CONTROL OF NONLINEAR SYSTEMS REPRESENTED IN QUASILINEAR FORM

Final Report
NASA Grant NAG-1-126

Josef A. Coetsee

December 1993

CONTROL OF NONLINEAR SYSTEMS REPRESENTED IN QUASILINEAR FORM

by

Josef Adriaan Coetsee

B.Sc.(Hons) Elek. Rand Afrikaans University, 1979
S.M. Aeronautics and Astronautics Massachusetts Institute of Technology, 1985

SUBMITTED TO THE DEPARTMENT OF
AERONAUTICS AND ASTRONAUTICS IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy
in Aeronautics and Astronautics
at the
Massachusetts Institute of Technology
February 1994

Signature of Author .....
Department of Aeronautics And Astronautics
December 17, 1993

Certified by ..... Wallace E. Vander Velde .....
Prof. W.E. Vander Velde, Thesis Supervisor
Department of Aeronautics and Astronautics

Certified by .....
Dr. H. Alexander, Thesis Committee Member
Senior Engineer, Orbital Sciences Corporation

Certified by ..... Jean-Jacques E. Slotine .....
Prof. J-J.E. Slotine, Thesis Committee Member
Department of Mechanical Engineering

Certified by ..... A. von Flotow .....
Dr. A. Von Flotow, Thesis Committee Member
President, Hood Technology Corporation

Accepted by .....
Prof. Harold Y. Wachman, Chairman
Department Graduate Committee

CONTROL OF NONLINEAR SYSTEMS REPRESENTED IN QUASILINEAR FORM

by

Josef Adriaan Coetsee

Submitted to the Department of Aeronautics And Astronautics
on December 17, 1993, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Aeronautics and Astronautics

Abstract

Methods to synthesize controllers for nonlinear systems are developed by exploiting the fact that under mild differentiability conditions, systems of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + G(\mathbf{x})\mathbf{u}$$

can be represented in quasilinear form, viz:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u}$$

Two classes of control methods are investigated:

- *zero-look-ahead control*, where the control input depends only on the current values of $A(\mathbf{x}), B(\mathbf{x})$. For this case the control input is computed by continuously solving a matrix Ricatti equation as the system progresses along a trajectory.
- *controllers with look-ahead*, where the control input depends on the future behavior of $A(\mathbf{x}), B(\mathbf{x})$. These controllers use the similarity between quasilinear systems, and linear time varying systems to find approximate solutions to optimal control type problems.

The methods that are developed are not *guaranteed* to be globally stable. However in simulation studies they were found to be useful alternatives for synthesizing control laws for a general class of nonlinear systems.

Prof. W.E. Vander Velde, Thesis Supervisor
Department of Aeronautics and Astronautics

Acknowledgments

I must first of all thank my wife and children for their support and the patience they have shown during the past few years. I also have to thank my parents, and my parents in law, for their very generous support and encouragement. Without them this endeavor would not have been possible.

Further thanks are due to my supervisor, Prof. Vander Velde. I can only speak in glowing terms of my experience of working with Prof. Vander Velde — he is a wonderful role model, both academically and personally. I also have to thank the other members of my doctoral committee, they have provided me with keen insights, advice and friendly support throughout.

This work conducted with support provided by NASA Langley under NASA Research Grant NAG-1-126.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Control engineering has played an important role in the development of industrial society. One of the earliest applications of control devices can be found in Watt's use of flyball governors to control the speed of steam engines [42]. Subsequently the discipline has undergone a great deal of theoretical development and has found practical application in a wide variety of systems such as the control of chemical processes, robotic manipulators, flight vehicles etc.

To develop control systems the following paradigm is often used:

1. It is assumed that there is a requirement for a system, also referred to as the plant, to behave in accordance with certain specifications. For example the requirement may be for a motor (the plant) to operate at a certain speed.

2. When the system's natural behavior does not in itself meet the requirements, it may be necessary to use some *control inputs* to induce the desired behavior. As an example some "regulator" may be required to ensure that a motor operates at the required speed.

3. The next step is to obtain an abstract dynamical model for the system. Usually it is assumed that the system's behavior is adequately described by a set of

ordinary differential equations. The differential equations are often derived on the basis of some physical principles, but other approaches are possible. For example it can simply be assumed that the system dynamics are described by a generic set of differential equations with unknown parameters. In this case the parameters in the dynamical model may be adjusted to ensure that the model reproduces available empirical data through a process of system identification. Alternatively the controller may be designed in such a way that it can cope with the uncertainty in the parameter values.

4. Once the dynamical model of the plant is available, one develops (usually by analytical means) a control law that will modify the plant's behavior to be in line with the original requirements.

5. Finally the controllers are implemented and tested to ensure that the overall requirements are met.

Much of control theory has been concerned with step 4, i.e. the synthesis of control laws. Ideally synthesis methods should:

- be generic (i.e. can be applied in a straightforward fashion to most systems),
- provide the designer with a convenient way to adjust the system's response, and
- guarantee stable system behavior.

Many design methods have been developed and these mostly meet the objectives above, depending on the *control objectives* and also the *class of system* that is to be controlled. We can make the following broad (but not exhaustive) classification:

A. Class of system:

– *Linear time invariant systems:* i.e. systems with dynamics of the form:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{1.1}$$

where $\mathbf{x}$ is the state vector and $\mathbf{u}$ is the set of control inputs.

- *Linear time varying systems:* i.e. systems with dynamics of the form:

$$\dot{\mathbf{x}} = A(t)\mathbf{x} + B(t)\mathbf{u} \tag{1.2}$$

Here the matrices $A(t), B(t)$ are known functions of time.

- *Nonlinear systems:* these are typically systems of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}), \text{ or} \tag{1.3}$$

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + G(\mathbf{x})\mathbf{u} \tag{1.4}$$

Within the class of nonlinear systems there exists a special class of systems, i.e. *globally feedback linearizable* systems. These systems are relatively easy to control — see Section 1.1.1 for more details.

- *Nonlinear time-varying systems:* i.e. systems with dynamics of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \tag{1.5}$$

B. Control objective:

- *Stabilization:* Here the only objective is to find a control law that ensures that the system is (globally) stable.
- *Optimization:* Here the objective is to find some control input that minimizes a cost function which typically has the form:

$$J = \mathbf{x}^T(t_f)H\mathbf{x}(t_f) + \int_{t_0}^{t_f}(\mathbf{x}^T Q\mathbf{x} + \mathbf{u}^T R\mathbf{u})\mathrm{d}\tau \tag{1.6}$$

(It is implicitly assumed that part of the control objective is to ensure that the system remains stable)

- *Adaptive Control:* In this case it is typically assumed that the system dynamics model contains some parameters, say $\mathbf{p}$, which are unknown to

the designer, for example:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{p}, \mathbf{u}) \tag{1.7}$$

The controller then adjusts itself to accommodate the unknown parameters $\mathbf{p}$ and still maintain overall stability.

- *Robust control:* Any of the above cases becomes a robust control problem if the objective is to maintain stability and performance objectives, despite the fact that assumptions regarding the nominal plant model may be incorrect.
- *Stochastic control:* If the system dynamics in any of the cases above are affected by a random process we have a stochastic control problem. An example would be the following linear time varying system that is driven by a random process $\mathbf{w}$:

$$\dot{\mathbf{x}} = A(t)\mathbf{x} + B(t)\mathbf{u} + \Gamma\mathbf{w} \tag{1.8}$$

Table 1.1 gives a (very) brief summary of the availability of synthesis methods for different control problems. We note that the control of general nonlinear systems that are not feedback linearizable remains a very difficult problem.

Our investigations will mostly be concerned with developing a synthesis method for general nonlinear systems which are not feedback linearizable. Since the global feedback linearization theory also illuminates the general nature of the problem we want to address, we will start by examining it. Following that, we will briefly discuss optimal control as an approach to synthesize control laws for nonlinear systems and then finally in Section 1.3 provide an overview of the methods to be developed in this thesis.

	Linear Time Invariant	Linear Time Varying	Non-linear (feedback linearizable)	Nonlinear (general)
Stabilization	Well developed [26]	Via opt. control [32]	Via LTI theory	Unresolved (possibly via opt control)
Optimization	Well developed [32]	Well developed [32]	Via LTI theory	Hard to apply.[15]
Adaptive Control	Well developed [19]	Partial results [43]	Ongoing [57],[58]	Unresolved
Robust Control	Well developed and ongoing [60],[16]	Via opt. control	Ongoing [55]	Unresolved (possibly via opt. control) [2]
Stochastic Control	Well developed [32]	Well developed [32]	Not fully developed	Hard to apply.[61]

Table 1.1: Availability of Synthesis Methods

1.1 Feedback Linearization of Nonlinear Systems

Global feedback linearization and normal forms for nonlinear systems are concepts that have recently been developed and provide much insight into the fundamental issues regarding the control of nonlinear systems. These concepts have their origins in work done by Krener [31] and Brockett [7] among others. Further developments in the theory were provided by Hirschorn [22], Jacubczyk and Respondek [25] as well as Hunt, Su and Meyer [23]. In discussing these ideas we will focus on single-input single-output systems, although the concepts can be generalized for multivariable systems (see e.g. Slotine and Li [59]).

1.1.1 Global Feedback Linearization

We first consider systems which are so-called *globally feedback linearizable*. Such systems are relatively easy to control since, in the words of Brockett, they are "... linear

systems in disguise ...". More precisely the system:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \tag{1.9}$$

is considered to be *globally feedback linearizable* if a combination of coordinate change:

$$\mathbf{z} = \varphi(\mathbf{x}) \tag{1.10}$$

and feedback:

$$u = \alpha(\mathbf{x}) + \beta(\mathbf{x})\nu \tag{1.11}$$

applied to (1.9) results in the linear time-invariant system:

$$\dot{\mathbf{z}} = A\mathbf{z} + \mathbf{b}\nu \tag{1.12}$$

where:

$$A = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ . & . & . & \dots & 0 & 0 \\ . & . & . & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ 1 \end{bmatrix} \tag{1.13}$$

for all $\mathbf{x}$. Technically it is required that:

- the transformation:

$$\mathbf{z} = \varphi(\mathbf{x}) \tag{1.14}$$

  be a *diffeomorphism*, i.e. $\varphi(.)$ must be smooth, and its inverse $\varphi^{-1}(.)$ must exist, and also be smooth.

- the functions $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ be smooth.

For a system of the type given in equation (1.9), precise conditions exist for us to determine whether a system is globally feedback linearizable or not (see e.g. Slotine

and Li [59], or Isidori [24]). In order to state these conditions compactly we use the following standard notation and definitions (see e.g. Slotine and Li [59]). Let $h(\mathbf{x})$ be a smooth scalar valued function, and $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ be smooth vector valued functions of the n-dimensional vector $\mathbf{x}$.

- The *gradient* of h w.r.t $\mathbf{x}$-coordinates is:

$$\nabla h \equiv \begin{bmatrix} \frac{\partial h}{\partial x_1} \\ \frac{\partial h}{\partial x_2} \\ \vdots \\ \frac{\partial h}{\partial x_n} \end{bmatrix} \tag{1.15}$$

- The *Jacobian* of $\mathbf{f}$ w.r.t to $\mathbf{x}$-coordinates is:

$$\nabla \mathbf{f} \equiv \begin{bmatrix} (\nabla f_1)^T \\ (\nabla f_2)^T \\ \vdots \\ (\nabla f_n)^T \end{bmatrix} \tag{1.16}$$

- The *Lie-derivative* of h w.r.t. $\mathbf{f}$ is given by:

$$L_{\mathbf{f}} h \equiv (\nabla h)^T \mathbf{f} \tag{1.17}$$

  and higher order Lie-derivatives are defined as:

$$L_{\mathbf{f}}^0 h = h \tag{1.18}$$

$$L_{\mathbf{f}}^i h = L_{\mathbf{f}} L_{\mathbf{f}}^{i-1} h \tag{1.19}$$

- The *Lie-bracket* of $\mathbf{f}$ and $\mathbf{g}$ is given by:

$$\mathrm{ad}_{\mathbf{f}}\mathbf{g} \equiv [\mathbf{f}, \mathbf{g}] \tag{1.20}$$

$$\equiv \nabla \mathbf{g}\, \mathbf{f} - \nabla \mathbf{f}\, \mathbf{g} \tag{1.21}$$

and higher order Lie-brackets are found from:

$$\mathrm{ad}_{\mathbf{f}}^{0}\mathbf{g} = \mathbf{g} \tag{1.22}$$

$$\mathrm{ad}_{\mathbf{f}}^{i}\mathbf{g} = \left[\mathbf{f}, \mathrm{ad}_{\mathbf{f}}^{i-1}\mathbf{g}\right] \tag{1.23}$$

- A linearly independent set of vector fields $\{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_m\}$ is said to be *involutive* if and only if there exist scalar functions $\alpha_{ijk}(\mathbf{x})$ such that:

$$[\mathbf{f}_i, \mathbf{f}_j] = \sum_{k=1}^{m} \alpha_{ijk}(\mathbf{x})\mathbf{f}_k(\mathbf{x}) \quad \forall i, j \tag{1.24}$$

With the definitions above, the necessary and sufficient conditions for feedback linearizability of (1.9) can be compactly stated. We have the following powerful theorem (see e.g. Slotine and Li [59] for details):

Theorem 1.1.1 *The nonlinear system (1.9) is globally feedback linearizable if and only if the following conditions hold for all* $\mathbf{x}$*:*

- *The vector fields*

$$\left\{\mathbf{g}, \mathrm{ad}_{\mathbf{f}}\mathbf{g}, \ldots, \mathrm{ad}_{\mathbf{f}}^{n-1}\mathbf{g}\right\} \tag{1.25}$$

  are linearly independent.

- *The vector fields*

$$\left\{\mathbf{g}, \mathrm{ad}_{\mathbf{f}}\mathbf{g}, \ldots, \mathrm{ad}_{\mathbf{f}}^{n-2}\mathbf{g}\right\} \tag{1.26}$$

  are involutive.

In proving the theorem above one also obtains a method to construct the coordinate transform and feedback law required to put (1.9) into the linear form of equations (1.12) and (1.13). The construction, which we describe next, requires one to solve a potentially difficult set of partial differential equations.

Assuming that conditions (1.25) and (1.26) are satisfied, the process to feedback linearize the system will be:

1. Solve the following set of p.d.e's for φ_1:

$$\nabla\varphi_1^T \mathrm{ad}_{\mathbf{f}}^i \mathbf{g} = 0 \quad i = 0, \ldots, n-2 \tag{1.27}$$

$$\nabla\varphi_1^T \mathrm{ad}_{\mathbf{f}}^{n-1} \mathbf{g} \neq 0 \tag{1.28}$$

2. Apply the coordinate transform:

$$\mathbf{z} = \boldsymbol{\varphi}(\mathbf{x}) = \begin{bmatrix} \varphi_1 \\ L_{\mathbf{f}}\varphi_1 \\ \vdots \\ L_{\mathbf{f}}^{n-1}\varphi_1 \end{bmatrix} \tag{1.29}$$

which will result in the following dynamics:

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \vdots \\ \dot{z}_n \end{bmatrix} = \begin{bmatrix} z_2 \\ z_3 \\ \vdots \\ \alpha(\mathbf{z}) + \beta(\mathbf{z})u \end{bmatrix} \tag{1.30}$$

where:

$$\alpha(\boldsymbol{\varphi}(\mathbf{x})) = L_{\mathbf{f}}^n \varphi_1 \tag{1.31}$$

$$\beta(\boldsymbol{\varphi}(\mathbf{x})) = L_{\mathbf{g}} L_{\mathbf{f}}^{n-1} \varphi_1 \tag{1.32}$$

We shall refer to (1.30) as the *cascade canonical form*[1] because the system is now made up of a string of integrators with the highest derivative driven by a nonlinear function of the state as well as the control input.

[1]The author is not aware of a standard term used for this canonical form for nonlinear systems

3. Apply the feedback:

$$u = \frac{1}{\beta(\boldsymbol{\varphi}(\mathbf{x}))}\left(-\alpha(\boldsymbol{\varphi}(\mathbf{x})) + \nu\right) \tag{1.33}$$

Once we have applied the feedback and transformations above we can apply the large body of control theory for linear systems. However the process outlined above is easier stated than applied. Difficulties arise on the following counts:

1. Establishing that a system is globally feedback linearizable (so that we know that the p.d.e's (1.28) have a solution) can require significant effort. For instance, if we cannot analytically check the conditions of Theorem 1.1.1, we have to test these conditions numerically at "every" point in the state space.

2. We have to solve the partial differential equations (1.28) which may be a far from trivial problem to do either analytically or numerically.

3. Once we have applied "inner loop" control (1.33), we expect to apply further state feedback control to get the L.T.I. system to behave appropriately, e.g.:

$$\nu = \mathbf{k}^T\mathbf{z} \tag{1.34}$$

$$= \begin{bmatrix} k_1, k_2, \ldots, k_n \end{bmatrix} \begin{bmatrix} \varphi_1 \\ L_{\mathbf{f}}\varphi_1 \\ \vdots \\ L_{\mathbf{f}}^{n-1}\varphi_1 \end{bmatrix} \tag{1.35}$$

For both the case where we can find $\mathbf{z} = \boldsymbol{\varphi}(\mathbf{x})$ analytically as well as the case where we find φ_1 numerically, the feedback law might be quite sensitive to errors in $\mathbf{x}$. Since the "outer loop" feedback, $\nu = \mathbf{k}^T\mathbf{z}$, involves derivatives of φ_1 we would expect any problems to be exacerbated if we have to find φ_1 by numerically solving the p.d.e.'s (1.28).

1.1.2 Partial Feedback Linearization and Normal Forms

In the previous section we outlined the situation where a system can be globally feedback linearized. As we might expect, there exists a large class of systems which cannot be feedback linearized, and thus remain difficult to control. To better understand the issues involved the concept of *partial feedback linearization* will be examined next. We shall see that if $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ are smooth enough, systems of the form (1.9) can be (locally) partially feedback linearized and put in a normal form.

Consider controllable systems of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \tag{1.36}$$

$$y = h(\mathbf{x}) \tag{1.37}$$

where $y = h(\mathbf{x})$ is an arbitrary smooth function of the system state. Usually y will be some output variable that we want to control. In its simplest form partial feedback linearization is achieved as follows: We successively differentiate y until the control input appears in the expression for say the rth derivative of y, viz:

$$\frac{\mathrm{d}^r y}{\mathrm{d}t^r} = \frac{\partial}{\partial \mathbf{x}}\left(\frac{\mathrm{d}^{r-1} y}{\mathrm{d}t^{r-1}}\right)(\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u) \tag{1.38}$$

$$= L_{\mathbf{f}}^r h + L_{\mathbf{g}} L_{\mathbf{f}}^{r-1} h \tag{1.39}$$

$$\equiv \alpha(\mathbf{x}) + \beta(\mathbf{x})u \tag{1.40}$$

The state equations for the system can then be written as:

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \vdots \\ \dot{y}_r \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} y_2 \\ y_3 \\ \vdots \\ \alpha(\mathbf{x}) + \beta(\mathbf{x})u \\ \boldsymbol{\xi}(\mathbf{x}, u) \end{bmatrix} \tag{1.41}$$

If $r = n$, where $n =$ is the dimension of the state vector, and we use the control:

$$u = \frac{1}{\beta(\mathbf{x})}\left(-\alpha(\mathbf{x}) + \nu\right) \tag{1.42}$$

we have globally feedback linearized the system. The process for global feedback linearization can thus be thought of as a method to appropriately construct $y = h(\mathbf{x})$, so that $r = n$, if it is at all possible. Furthermore, the conditions of Theorem 1.1.1 can then be thought of as tests to determine whether a suitable $h(\mathbf{x})$ can be found.

In equation (1.41) we see that $\psi(\mathbf{x}, u)$ depends on both $\mathbf{x}$ and u. It can be shown that using an appropriate transformation (which is also constructed by solving partial differential equations) the system can be put in the following normal form:

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \vdots \\ \dot{y}_r \\ \dot{\boldsymbol{\eta}} \end{bmatrix} = \begin{bmatrix} y_2 \\ y_3 \\ \vdots \\ \alpha(\mathbf{y}, \boldsymbol{\eta}) + \beta(\mathbf{y}, \boldsymbol{\eta})u \\ \mathbf{w}(\mathbf{y}, \boldsymbol{\eta}) \end{bmatrix} \tag{1.43}$$

This normal form enables us to better understand what we can achieve in controlling the system. For instance, if $\beta(\mathbf{y}, \boldsymbol{\eta}) \neq 0 \ \forall \mathbf{x}$, we can fully determine the input-output relation between u and y. However, since $\mathbf{y}$ acts as an input to the $\boldsymbol{\eta}$ states, we see that y following a certain trajectory forces a specific behavior for the internal dynamics:

$$\dot{\boldsymbol{\eta}} = \mathbf{w}(\mathbf{y}, \boldsymbol{\eta}) \tag{1.44}$$

Clearly it is possible that the states $\boldsymbol{\eta}$, which are not observable via $h(\mathbf{x})$, can become unstable, depending on the $\mathbf{y}$ trajectory.

An important special case occurs when $\mathbf{y} \equiv \mathbf{0}$. The dynamics of (1.43) subject to the constraint $\mathbf{y} \equiv \mathbf{0}$ are called *zero-dynamics* [59] of the system. In terms of the normal

form we get:

$$\begin{bmatrix} \dot{\mathbf{y}} \\ \dot{\boldsymbol{\eta}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{w}(\mathbf{0}, \boldsymbol{\eta}) \end{bmatrix} \tag{1.45}$$

If the zero-dynamics of the system is asymptotically stable the nonlinear system is said to be *asymptotically minimum phase*. It can be shown that this definition captures our standard understanding of minimum-phase systems for the case where we are dealing with linear systems (see Slotine and Li [59]).

At this point it becomes clear what the challenge is regarding the control of nonlinear systems. Typically the problem will be to obtain certain desirable behavior from our output variable y while still maintaining stability of the internal dynamics.

1.2 Optimal Control

Another approach to synthesizing controllers for nonlinear systems is to find the control input through solving an optimal control problem. In principle this approach meets all our objectives for an ideal synthesis method. For example if our objective is to stabilize a nonlinear system of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \quad \text{with equilibrium point:} \tag{1.46}$$

$$\mathbf{0} = \mathbf{f}(\mathbf{0}, \mathbf{0}) \tag{1.47}$$

We then pose an optimization problem which has the objective to minimize the following cost function:

$$J = \int_{t_0}^{\infty} (\mathbf{x}^T Q \mathbf{x} + \mathbf{u}^T R \mathbf{u}) \mathrm{d}\tau \tag{1.48}$$

where we assume that both Q and R are positive definite matrices. If the system (1.46) is controllable [63] and we apply the control input that minimizes (1.48) the system will be stable as the following standard argument shows:

Since (1.46) is controllable there exists some control input that will drive the system

to the origin from any initial condition. If we apply this control input to the system we will incur a certain finite cost, say $J_c(\mathbf{x}_0)$. Furthermore if we apply the optimal control input to the system there will be a certain associated cost, say $\bar{J}(\mathbf{x}_0)$ which has to be less than J_c and thus finite. Now note that since Q is positive definite the cost incurred for an unstable response would be infinite. Hence the optimal input cannot cause unstable responses.

A further attractive aspect of the optimal control approach is that it provides us with a method to conveniently adjust the system responses by adjusting the cost function.

The only undesirable feature of the optimal control approach is that it is computationally very demanding. If the goal is to obtain the optimal control as a state feedback law we have to solve a nonlinear partial differential equation, the Hamilton-Jacobi-Bellman equation , for the whole region of the state space where the system is to operate (see Section 3.2.1 for further discussion). Alternatively we have to find the optimal control input by numerically solving a two point boundary value problem [15].

1.3 Overview

As indicated earlier we will be mostly concerned with developing a controller synthesis method for nonlinear systems of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + G(\mathbf{x})\mathbf{u} \tag{1.49}$$

We will exploit the fact (see Chapter 2) that provided $\mathbf{f}(\mathbf{x})$ meets certain smoothness requirements, equation (1.49) can be written in the following quasilinear form:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{1.50}$$

In Chapter 3 we will investigate a design methodology where the control input is com-

puted by continuously solving a matrix Ricatti equation based on the instantaneous properties of $A(\mathbf{x}), B(\mathbf{x})$. We shall refer to this as a control law with zero-look-ahead. Related work appears in the thesis by Shamma [53] who investigates gain scheduling in the context of the LQG\LTR methodology [60], and takes a different perspective with regard to the stability issue.

In Chapter 4 we develop control laws that take into account the "future behavior" of $A(\mathbf{x}), B(\mathbf{x})$. We refer to these as control laws with look-ahead. The methods we investigate will all have underlying optimal control problems that have to be solved. By exploiting the quasilinear form we will obtain approximate solutions to these optimal control problems.

Finally in Chapter 5 we will apply the controller synthesis methods we developed to control a complex nonlinear system, i.e. a flexible robotic manipulator similar to the space shuttle Remote Manipulator System [20].

Chapter 2

Quasilinear Description

In this chapter we will examine some of the basic issues in dealing with nonlinear dynamical systems of the form:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{2.1}$$

We shall refer to such systems as *quasilinear* systems.

2.1 Generality of the Quasilinear Form

The following lemma from Vidyasagar [63] shows that under mild differentiability conditions, systems of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u} \tag{2.2}$$

can always be transformed to the quasilinear form of equation (2.1). The lemma is more general than we will need since it caters for the case where $\mathbf{f}(\mathbf{0}) \neq \mathbf{0}$, whereas we will generally assume $\mathbf{f}(\mathbf{0}) = \mathbf{0}$.

Lemma 2.1.1 *Suppose* $\mathbf{f} : R^n \rightarrow R^n$ *is continuously differentiable. Then there exists a continuous function* $A : R^n \rightarrow R^{n \times n}$ *such that:*

$$\mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{0}) + A(\mathbf{x})\mathbf{x}, \; \forall \mathbf{x} \in R^n \tag{2.3}$$

Proof:

Fix $\mathbf{x}$ and consider $\mathbf{f}(\lambda \mathbf{x})$ as a function of the scalar parameter λ. Then

$$\begin{aligned} \mathbf{f}(\mathbf{x}) &= \mathbf{f}(\mathbf{0}) + \int_0^1 \frac{d}{d\lambda} \mathbf{f}(\lambda \mathbf{x}) \, d\lambda & (2.4) \\ &= \mathbf{f}(\mathbf{0}) + \left[\int_0^1 \nabla_{\mathbf{x}} \mathbf{f}(\lambda \mathbf{x}) \, d\lambda \right] \mathbf{x} & (2.5) \end{aligned}$$

so that

$$A(\mathbf{x}) \equiv \int_0^1 \nabla_{\mathbf{x}} \mathbf{f}(\lambda \mathbf{x}) \, d\lambda \tag{2.6}$$

———*Q.E.D.*

The above formula gives us a *specific* quasilinear form for any C^1 function $\mathbf{f}(\mathbf{x})$. Equation (2.6) is useful in cases where the quasilinear form may not be immediately apparent. For example let:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \sin(x_1 + x_2) \\ \sin(x_2) \end{bmatrix} \tag{2.7}$$

Equation (2.6) gives:

$$A(\mathbf{x}) = \begin{bmatrix} \frac{\sin(x_1+x_2)}{(x_1+x_2)} & \frac{\sin(x_1+x_2)}{(x_1+x_2)} \\ 0 & \frac{\sin(x_2)}{x_2} \end{bmatrix} \tag{2.8}$$

Note that, in general, the quasilinear form for a given function $\mathbf{f}(\mathbf{x})$ is not unique. In fact, we see that we can add any vector that is orthogonal to $\mathbf{x}$, to any row of $A(\mathbf{x})$, and still get the same value for $A(\mathbf{x})\mathbf{x}$. For the example above we see that an

alternative quasilinear form for $\mathbf{f}(\mathbf{x})$ is:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \frac{\sin(x_1+x_2)}{x_1+x_2} - k_1 x_2 & \frac{\sin(x_1+x_2)}{x_1+x_2} + k_1 x_1 \\ -k_2 x_2 & \frac{\sin(x_2)}{x_2} + k_2 x_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{2.9}$$

This non-uniqueness can be exploited as we shall see in Section 2.3.2.

2.2 Quasilinear Form for Mechanical Systems

In this section we will show that the equations of motion of a class of mechanical systems which operate in the absence of gravity can be put into quasilinear form without solving the potentially difficult integrals of equation (2.6). (See also Appendix A which describes a set of computer routines that utilize this approach to automatically derive quasilinear equations of motion for flexible manipulators.)

We will use Lagrange's equations. They are:

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_i}) - \frac{\partial L}{\partial q_i} = Q_i, \quad i = 1 \ldots n \tag{2.10}$$

where:

n = the number of degrees of freedom in the system

q_i = ith generalized coordinate

Q_i = ith generalized force

$L = T - U$

T = kinetic energy

U = potential energy

If we can express the kinetic and potential energy of the system in the form:

$$T = \frac{1}{2}\dot{\mathbf{q}}^T H(\mathbf{q})\dot{\mathbf{q}} \tag{2.11}$$

$$U = \frac{1}{2}\mathbf{q}^T K \mathbf{q} \tag{2.12}$$

where:

$\mathbf{q}^T = [q_1, q_2, \ldots, q_n]$ is a vector of generalized coordinates

$H(\mathbf{q})$ = a generalized inertia matrix

K = a generalized stiffness matrix

then Lagrange's equations can be written in vector format as:

$$H\ddot{\mathbf{q}} + \dot{H}\dot{\mathbf{q}} - \nabla_{\mathbf{q}} T + K\mathbf{q} = \mathbf{Q} \tag{2.13}$$

where: $\mathbf{Q} = [Q_1, Q_2, \ldots, Q_n]^T$ is a vector of generalized forces, and:

$$\nabla_{\mathbf{q}} T \equiv \frac{\partial}{\partial \mathbf{q}} T \tag{2.14}$$

$$\equiv \left[\frac{\partial T}{\partial q_1}, \frac{\partial T}{\partial q_2}, \ldots, \frac{\partial T}{\partial q_n}\right]^T \tag{2.15}$$

Now:

$$\nabla_{\mathbf{q}} T = \left[\dot{\mathbf{q}}^T \left(\frac{\partial}{\partial \mathbf{q}} \left(H(\mathbf{q})\dot{\mathbf{q}}\right)\right)\right]^T \tag{2.16}$$

so that the equations of motion can be expressed as:

$$H\ddot{\mathbf{q}} + \left(\dot{H} - \left[\frac{\partial}{\partial \mathbf{q}} H(\mathbf{q})\dot{\mathbf{q}}\right]^T\right)\dot{\mathbf{q}} + K\mathbf{q} = \mathbf{Q} \tag{2.17}$$

or

$$H(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + K\mathbf{q} = \mathbf{Q} \tag{2.18}$$

To get a state space description we furthermore assume that the system is controlled via the generalized forces and that they can be expressed as:

$$\mathbf{Q} = M(\mathbf{q}, \dot{\mathbf{q}})\mathbf{u} \tag{2.19}$$

where the components of $\mathbf{u}$ are the control inputs to the system. We then get the

desired state space form:

$$\begin{bmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} 0 & I \\ -H(\mathbf{q})^{-1}K & -H(\mathbf{q})^{-1}C(\mathbf{q},\dot{\mathbf{q}}) \end{bmatrix} \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix} + \begin{bmatrix} 0 \\ H(\mathbf{q})^{-1}M(\mathbf{q},\dot{\mathbf{q}}) \end{bmatrix} \mathbf{u} \quad (2.20)$$

Note that the key to getting to the quasilinear form is to express the kinetic and potential energy as in equations (2.11) and (2.12). In the following example we shall use a double pendulum system to show how we can find the desired form for the kinetic and potential energy. We shall see that it is generally relatively easy to find the desired form for the kinetic energy. On the other hand, the expression for potential energy will only be in the desired form if we are dealing with the equivalent of "linear springs". This will typically be the case for flexible mechanical space based systems. If we are dealing with nonlinear "springs", e.g. gravity, we generally will have to use equation (2.6).

Example 2.2.1

Figure (2-1) shows a double pendulum with rigid massless links of lengths l_1 and l_2 respectively. The links also have tip masses m_1 and m_2 and rotational springs K_1 and K_2 attached. For this system the generalized coordinates are θ_1 and θ_2, viz:

$$\mathbf{q} \equiv \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} \quad (2.21)$$

We first find an expression for the kinetic energy. Let $\mathbf{R}_1$ and $\mathbf{R}_2$ be position vectors to each of the masses. Then kinetic energy of the system is given by:

$$T = \frac{m_1}{2}\left(\frac{\mathrm{d}\mathbf{R}_1}{\mathrm{d}t} \cdot \frac{\mathrm{d}\mathbf{R}_1}{\mathrm{d}t}\right) + \frac{m_2}{2}\left(\frac{\mathrm{d}\mathbf{R}_2}{\mathrm{d}t} \cdot \frac{\mathrm{d}\mathbf{R}_2}{\mathrm{d}t}\right) \quad (2.22)$$

with:

$$\mathbf{R}_1 = \begin{bmatrix} l_1\cos(\theta_1) \\ l_1\sin(\theta_1) \end{bmatrix}, \quad \mathbf{R}_2 = \begin{bmatrix} l_1\cos(\theta_1) + l_2\cos(\theta_1+\theta_2) \\ l_2\sin(\theta_1) + l_2\sin(\theta_1+\theta_2) \end{bmatrix} \quad (2.23)$$

For systems not made up of point masses we can simply express the kinetic energy

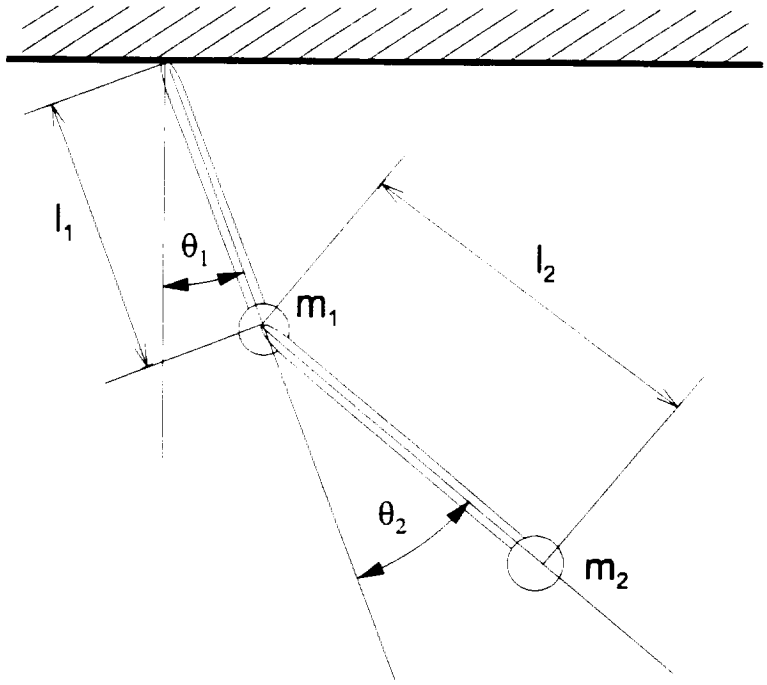


Figure 2-1: Double Pendulum

as an integral over infinitesimal mass elements viz:

$$T = \int_{\text{vol}} \frac{\mathrm{d}\mathbf{R}}{\mathrm{d}t} \cdot \frac{\mathrm{d}\mathbf{R}}{\mathrm{d}t} \mathrm{dm} \tag{2.24}$$

For the double pendulum system we get:

$$\frac{\mathrm{d}\mathbf{R}_i}{\mathrm{d}t} = \left[\begin{array}{cc} \frac{\partial}{\partial\theta_1}\mathbf{R}_i & \frac{\partial}{\partial\theta_2}\mathbf{R}_i \end{array} \right] \left[\begin{array}{c} \dot{\theta}_1 \\ \dot{\theta}_2 \end{array} \right] \tag{2.25}$$

$$\equiv J_i(\mathbf{q})\dot{\mathbf{q}} \tag{2.26}$$

where:

$$J_1(\mathbf{q}) = \left[\begin{array}{cc} -l_1 \sin(\theta_1) & 0 \\ l_1 \cos(\theta_1) & 0 \end{array} \right] \tag{2.27}$$

$$J_2(\mathbf{q}) = = \begin{bmatrix} -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix} \tag{2.28}$$

Now we can express the kinetic energy in the desired form:

$$\begin{aligned} T &= \frac{m_1}{2}\dot{\mathbf{q}}^T J_1^T J_1 \dot{\mathbf{q}} + \frac{m_2}{2}\dot{\mathbf{q}}^T J_2^T J_2 \dot{\mathbf{q}} && (2.29) \\ &= \frac{1}{2}\dot{\mathbf{q}}^T \left[m_1 J_1^T J_1 + m_2 J_2^T J_2 \right] \dot{\mathbf{q}} && (2.30) \\ &\equiv \frac{1}{2}\dot{\mathbf{q}}^T H(\mathbf{q}) \dot{\mathbf{q}} && (2.31) \end{aligned}$$

Next we find an expression for the potential energy. In the absence of gravity the potential energy of the system is given by:

$$\begin{aligned} U_{spring} &= \frac{1}{2}\left(K_1 \theta_1^2 + K_2 \theta_2^2 \right) && (2.32) \\ &= \frac{1}{2}\mathbf{q}^T \begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix} \mathbf{q} && (2.33) \\ &\equiv \frac{1}{2}\mathbf{q}^T K \mathbf{q} && (2.34) \end{aligned}$$

which is of the desired form viz. equation (2.12). We achieved this form because we had linear springs. If we include the effects of gravity we will not be able to conveniently express the potential energy in the form of equation (2.12) - gravity does not act like a linear spring. Nevertheless we can still obtain a quasilinear form for our dynamics when gravity is present by e.g. using (2.6) as we shall now show.

The potential energy due to gravity is given by:

$$U_{gravity} = -m_1 g l_1 \cos(\theta_1) - m_2 g \left(l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) \right) \tag{2.35}$$

This gives rise to the following additional term in equation (2.13):

$$\nabla_{\mathbf{q}} U_{gravity} = \begin{bmatrix} m_1 g l_1 \sin(\theta_1) - m_2 g \left(-l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2) \right) \\ m_2 g l_2 \sin(\theta_1 + \theta_2) \end{bmatrix} \tag{2.36}$$

In this case we can express the gravity term in quasilinear form as follows:

$$\nabla_{\mathbf{q}} U_{gravity} = \begin{bmatrix} \frac{(m_1 l_1 + m_2 l_1) g \sin(\theta_1)}{\theta_1} + \frac{m_2 g l_2 \sin(\theta_1+\theta_2)}{(\theta_1+\theta_2)} & \frac{m_2 g l_2 \sin(\theta_1+\theta_2)}{(\theta_1+\theta_2)} \\ \frac{m_2 g l_2 \sin(\theta_1+\theta_2)}{(\theta_1+\theta_2)} & \frac{m_2 g l_2 \sin(\theta_1+\theta_2)}{(\theta_1+\theta_2)} \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} \quad (2.37)$$

$$\equiv G_{grav}(\mathbf{q})\mathbf{q} \quad (2.38)$$

We then get a result similar to equation (2.17):

$$H\ddot{\mathbf{q}} + \left(\dot{H} - \left[\frac{\partial}{\partial \mathbf{q}} H(\mathbf{q})\dot{\mathbf{q}} \right]^T \right) \dot{\mathbf{q}} + \left[K + G_{grav}(\mathbf{q}) \right] \mathbf{q} = \mathbf{0} \quad (2.39)$$

The rest of the process follows as outlined above. The resulting quasilinear equations of motion are given in Appendix B.

Remarks:

- The quasilinear form of equation (2.20) is already in a partially feedback linearized form, although not the normal form of (1.43). However, we are using "natural" coordinates, i.e. the variables of the equations of motion have physical significance, whereas the variables $\boldsymbol{\eta}$ in the normal form of (1.43) usually do not have physical significance.
- If the number of actuators is equal to the number of degrees of freedom of the system, the quasilinear equations will be in the desirable *cascade canonical form* of equation (1.30). This will, for instance, be the case if we are dealing with robots with rigid links with actuators at each joint.

2.3 Stability Analysis Using the Properties of $A(\mathbf{x})$

In this section we will examine some stability tests for the ordinary differential equation (o.d.e.):

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} \quad (2.40)$$

that are based *only* on the properties of the matrix $A(\mathbf{x})$. Because the o.d.e. (2.40) represents a very large class of nonlinear systems, viz. any o.d.e. $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ with $\mathbf{f}(.) \in C^1$, we do not expect to find stability tests that are easy to apply and at the same time provide both *necessary* and sufficient conditions for stability. We note that the tests we discuss will also be useful for analyzing the stability of linear time varying systems where:

$$\dot{\mathbf{x}} = A(t)\mathbf{x} \tag{2.41}$$

2.3.1 Stability Analysis Based on Eigenstructure of $A(\mathbf{x})$

One is tempted to predict the stability of (2.40) based on the eigenvalues of $A(\mathbf{x})$. We will show that the eigenvalues do not provide us with enough information to deduce stability and that we have to take into account the rate of change of the eigenstructure of $A(\mathbf{x})$. The results we present are similar in spirit to other results regarding the stability of slowly varying systems (see e.g. [43],[63]).

We have

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} \tag{2.42}$$

Assume that all the eigenvalues of $A(\mathbf{x})$ are distinct, real and strictly negative. Then:

$$A(\mathbf{x}) = T(\mathbf{x})\Lambda(\mathbf{x})T^{-1}(\mathbf{x}) \tag{2.43}$$

where

$T(\mathbf{x})$ = a matrix containing the eigenvectors of $A(\mathbf{x})$

$\Lambda(\mathbf{x}) = \text{diag}(\lambda_1(\mathbf{x}), \lambda_2(\mathbf{x}), \ldots, \lambda_n(\mathbf{x}))$ is a diagonal matrix containing the eigenvalues of $A(\mathbf{x})$ with $\lambda_i(\mathbf{x}) \leq -\epsilon < 0 \ \forall \mathbf{x}, \ i = 1 \ldots n$.

We can then make the coordinate change:

$$\mathbf{z} = T^{-1}(\mathbf{x})\mathbf{x} \tag{2.44}$$

which results in the following dynamics for the transformed system:

$$\begin{aligned} \dot{\mathbf{z}} &= T^{-1}(\mathbf{x})\dot{\mathbf{x}} + \frac{\mathrm{d}}{\mathrm{d}t}\left(T^{-1}(\mathbf{x})\right)\mathbf{x} && (2.45)\\ &= T^{-1}A(\mathbf{x})\left(TT^{-1}\right)\mathbf{x} - T^{-1}\dot{T}T^{-1}\mathbf{x} && (2.46)\\ &= \Lambda(\mathbf{z})\mathbf{z} - \left(T^{-1}(\mathbf{z})\dot{T}(\mathbf{z})\right)\mathbf{z} && (2.47) \end{aligned}$$

where for notational convenience we have not shown the state dependence of T in equation (2.46). To determine the stability of the system we use the following simple observation:

Fact 2.3.1 *The system of o.d.e.'s:*

$$\dot{\mathbf{z}} = \Lambda(\mathbf{z})\mathbf{z} + M(\mathbf{z})\mathbf{z} \tag{2.48}$$

with:

$$\Lambda(\mathbf{z}) = \begin{bmatrix} \lambda_1(\mathbf{z}) & 0 & \dots & 0 \\ 0 & \lambda_2(\mathbf{z}) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n(\mathbf{z}) \end{bmatrix} \tag{2.49}$$

will be asymptotically stable provided that:

$$\mathbf{z}^T\Lambda(\mathbf{z})\mathbf{z} + \mathbf{z}^T M(\mathbf{z})\mathbf{z} < 0 \;\; \forall \mathbf{z} \neq 0 \tag{2.50}$$

Proof:

Use the candidate Lyapunov function:

$$V(\mathbf{z}) = \frac{1}{2}\mathbf{z}^T\mathbf{z} \tag{2.51}$$

which is positive definite and radially unbounded. Then:

$$\dot{V} = \mathbf{z}^T \dot{\mathbf{z}} \tag{2.52}$$

$$= \mathbf{z}^T \Lambda(\mathbf{z})\mathbf{z} + \mathbf{z}^T M(\mathbf{z})\mathbf{z} \tag{2.53}$$

and global asymptotic stability follows since $\dot{V} < 0 \ \ \forall \mathbf{z} \neq 0$.

Q.E.D.

We can apply this fact to our case by noting that:

$$\mathbf{z}^T \Lambda(\mathbf{z})\mathbf{z} \leq -\epsilon \mathbf{z}^T \mathbf{z} \tag{2.54}$$

and

$$\mathbf{z}^T T^{-1}(\mathbf{z})\dot{T}(\mathbf{z})\mathbf{z} \leq \|\mathbf{z}\|_2 \, \|T^{-1}(\mathbf{z})\dot{T}(\mathbf{z})\mathbf{z}\|_2 \tag{2.55}$$

$$\leq \|T^{-1}(\mathbf{z})\dot{T}(\mathbf{z})\|_2 \, \mathbf{z}^T \mathbf{z} \tag{2.56}$$

so that if:

$$\|T^{-1}(\mathbf{z})\dot{T}(\mathbf{z})\|_2 < \nu < \epsilon \tag{2.57}$$

we have:

$$\mathbf{z}^T \Lambda(\mathbf{z})\mathbf{z} + \mathbf{z}^T T^{-1}\dot{T}\mathbf{z} < (-\epsilon + \nu)\, \mathbf{z}^T \mathbf{z} < 0 \tag{2.58}$$

That is if the rate of change of the eigenstructure is small enough, viz. $\dot{T} \approx 0$, we can infer stability pointwise from the eigenvalues of $A(\mathbf{x})$. For example, consider the following seemingly complex second order nonlinear system:

$$\begin{aligned} \dot{x_1} = & -(15\, x_2^9 + 1260\, x_1^6 x_2^3 + 1890\, x_1^5 x_2^4 + 1890\, x_1^4 x_2^5 \\ & +1260\, x_1^3 x_2^6 + 540\, x_1^2 x_2^7 + 135\, x_1\, x_2^8 \\ & +135\, x_1^8 x_2 + 540\, x_1^7 x_2^2 + 51\, x_2 + 54\, x_1 + 15\, x_1^9 \\ & -6\, x_1^2 - 15\, x_1\, x_2 - 9\, x_2^2 - 27\, x_1^3 x_2 - 45\, x_1^2 x_2^2 \end{aligned}$$

$$
\begin{aligned}
& -33\,x_1\,x_2^3 + 461\,x_1^2 x_2 + 550\,x_1\,x_2^2 + 432\,x_1^4 x_2 \\
& +1026\,x_1^3 x_2^2 + 1206\,x_1^2 x_2^3 + 702\,x_1\,x_2^4 - 6\,x_1^4 \\
& -9\,x_2^4 + 130\,x_1^3 + 219\,x_2^3 + 72\,x_1^5 + 162\,x_2^5 + 15\,x_1^7 \\
& +15\,x_2^7 + 105\,x_1^6 x_2 + 315\,x_1^5 x_2^2 + 525\,x_1^4 x_2^3 + 525\,x_1^3 x_2^4 + \\
& 315\,x_1^2 x_2^5 + 105\,x_1\,x_2^6)/(1 + x_1^2 + 2\,x_1\,x_2 + x_2^2)
\end{aligned} \tag{2.59}
$$

$$
\begin{aligned}
\dot{x_2} \;=\; & (10\,x_2^9 + 840\,x_1^6 x_2^3 + 1260\,x_1^5 x_2^4 + 1260\,x_1^4 x_2^5 \\
& +840\,x_1^3 x_2^6 + 360\,x_1^2 x_2^7 + 90\,x_1\,x_2^8 + 90\,x_1^8 x_2 \\
& +360\,x_1^7 x_2^2 + 31\,x_2 + 34\,x_1 + 10\,x_1^9 - 4\,x_1^2 - 10\,x_1\,x_2 \\
& -6\,x_2^2 - 18\,x_1^3 x_2 - 30\,x_1^2 x_2^2 - 22\,x_1\,x_2^3 + 305\,x_1^2 x_2 \\
& +364\,x_1\,x_2^2 + 288\,x_1^4 x_2 + 684\,x_1^3 x_2^2 + 804\,x_1^2 x_2^3 \\
& +468\,x_1\,x_2^4 - 4\,x_1^4 - 6\,x_2^4 + 86\,x_1^3 + 145\,x_2^3 + 48\,x_1^5 \\
& +108\,x_2^5 + 10\,x_1^7 + 10\,x_2^7 + 70\,x_1^6 x_2 + 210\,x_1^5 x_2^2 + 350\,x_1^4 x_2^3 \\
& +350\,x_1^3 x_2^4 + 210\,x_1^2 x_2^5 + 70\,x_1\,x_2^6)/(1 + x_1^2 + 2\,x_1\,x_2 + x_2^2)
\end{aligned} \tag{2.60}
$$

These equations can be written in quasilinear form as:

$$
\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} \tag{2.61}
$$

where

$$
\begin{aligned}
A(1,1) \;=\; & -\Big\{54 + 58\,(x_1 + x_2)^2 + 18\,(-2\,x_1 - 3\,x_2)^2 \\
& +18\,(-2\,x_1 - 3\,x_2)^2\,(x_1 + x_2)^2 \\
& -6\,x_1 - 9\,x_2 + 3\,(-2\,x_1 - 3\,x_2)\,(x_1 + x_2)^2 \\
& +15\,(x_1 + x_2)^6 + 15\,(x_1 + x_2)^8\Big\} \,/\, \Big\{1 + (x_1 + x_2)^2\Big\}
\end{aligned} \tag{2.62}
$$

$$
\begin{aligned}
A(1,2) \;=\; & -\Big\{51 + 57\,(x_1 + x_2)^2 + 18\,(-2\,x_1 - 3\,x_2)^2 \\
& +18\,(-2\,x_1 - 3\,x_2)^2\,(x_1 + x_2)^2 \\
& -6\,x_1 - 9\,x_2 + 3\,(-2\,x_1 - 3\,x_2)\,(x_1 + x_2)^2
\end{aligned}
$$

$$+15\,(x_1+x_2)^6+15\,(x_1+x_2)^8\Big\}\Big/\Big\{1+(x_1+x_2)^2\Big\} \tag{2.63}$$

$$\begin{aligned} A(2,1) &= \Big\{34+38\,(x_1+x_2)^2+12\,(-2\,x_1-3\,x_2)^2 \\ &\quad +12\,(-2\,x_1-3\,x_2)^2\,(x_1+x_2)^2-4\,x_1-6\,x_2 \\ &\quad +2\,(-2\,x_1-3\,x_2)\,(x_1+x_2)^2+10\,(x_1+x_2)^6 \\ &\quad +10\,(x_1+x_2)^8\Big\}\Big/\Big\{1+(x_1+x_2)^2\Big\} \end{aligned} \tag{2.64}$$

$$\begin{aligned} A(2,2) &= \Big\{31+37\,(x_1+x_2)^2+12\,(-2\,x_1-3\,x_2)^2 \\ &\quad +12\,(-2\,x_1-3\,x_2)^2\,(x_1+x_2)^2-4\,x_1 \\ &\quad -6\,x_2+2\,(-2\,x_1-3\,x_2)\,(x_1+x_2)^2+10\,(x_1+x_2)^6 \\ &\quad +10\,(x_1+x_2)^8\Big\}\Big/\Big\{1+(x_1+x_2)^2\Big\} \end{aligned} \tag{2.65}$$

For $A(\mathbf{x})$ above, an eigenvector matrix is:

$$\begin{bmatrix} -1 & -3/2 \\ 1 & 1 \end{bmatrix} \tag{2.66}$$

Since eigenvectors are only uniquely determined up to a scale factor we use the following (eigenvector) transformation matrix:

$$T=\begin{bmatrix} -\alpha & -\frac{3}{2}\beta \\ \alpha & \beta \end{bmatrix} \tag{2.67}$$

where we assume that α and β are constants. For these eigenvectors the eigenvalues of $A(\mathbf{x})$ are given by:

$$\lambda_1(\mathbf{x}) = -\frac{3+x_1^2+2\,x_1\,x_2+x_2^2}{1+x_1^2+2\,x_1\,x_2+x_2^2} \tag{2.68}$$

$$\begin{aligned} \lambda_2(\mathbf{x}) &= \Big(-5\,x_2^6-30\,x_1\,x_2^5-75\,x_1^2x_2^4-100\,x_1^3x_2^3-54\,x_2^2-75\,x_1^4x_2^2 \\ &\quad -30\,x_1^5x_2+3\,x_2-72\,x_1\,x_2-24\,x_1^2-20+2\,x_1-5\,x_1^6\Big) \end{aligned} \tag{2.69}$$

Then upon applying the state transform:

$$\mathbf{z} = T^{-1}\mathbf{x} \tag{2.70}$$

and noting that $\dot{T} = 0$ we get:

$$\dot{\mathbf{z}} = \Lambda(\mathbf{z})\mathbf{z} \tag{2.71}$$

with:

$$\Lambda(\mathbf{z}) = \begin{bmatrix} -(12 + \beta^2 z_2^2)/(4 + \beta^2 z_2^2) & 0 \\ 0 & -20 + \alpha\, z_1 - 6\,\alpha^2 z_1^2 - \frac{5}{64}\,\beta^6 z_2^6 \end{bmatrix} \tag{2.72}$$

We see that $\lambda_1(\mathbf{z}) < 0$, $\lambda_2(\mathbf{z}) < 0$ and also $\dot{T} = 0$ and hence conclude that the system is stable.

2.3.2 Stability Analysis via Matrix Measure Theorems

Another approach to inferring stability of (2.40) from the properties of $A(\mathbf{x})$ is to use the *matrix measure* or *logarithmic derivative* of $A(\mathbf{x})$ which is given by:

Definition 2.3.1 *Let A be an $n \times n$ matrix and $\|.\|_i$ be an induced norm on $R^{n\times n}$. The corresponding matrix measure $\mu(A)$ of the matrix A is given by:*

$$\mu(A) \equiv \lim_{\epsilon \to 0+} \frac{\|I + \epsilon A\|_i - 1}{\epsilon} \tag{2.73}$$

The matrix measure was originally used to determine stability and error bounds for numerical integration methods [13] and later used by Coppel [12] to obtain solution bounds for linear time varying o.d.e.'s. Desoer and Haneda [14] also used matrix measures to bound solutions of o.d.e.'s, bound accumulated truncation errors for numerical integration schemes and determine convergence regions for the Newton-Raphson solutions of nonlinear equations. Further related work appears in [51], [52]

and [44].

The matrix measure is defined so as to easily bound the behavior of $\|\mathbf{x}(t)\|$ for o.d.e.'s of the form:

$$\dot{\mathbf{x}} = A(\mathbf{x}, t)\mathbf{x} \tag{2.74}$$

The following theorem shows how this is done.

Theorem 2.3.1 *[63],[14] Consider the o.d.e.:*

$$\dot{\mathbf{x}} = A(\mathbf{x}, t)\mathbf{x}, \quad \mathbf{x} \in R^n \tag{2.75}$$

Let $\|.\|$ be a norm on R^n, and $\|.\|_i$ and $\mu(.)$ be the corresponding induced norm and matrix measure on $R^{n\times n}$ respectively. Then:

$$\frac{d^+}{\mathrm{dt}}\|\mathbf{x}(t)\| \leq \mu\left(A(\mathbf{x}, t)\right)\|\mathbf{x}(t)\| \tag{2.76}$$

Suppose furthermore that there exist continuous functions $\alpha(t), \beta(t)$ such that:

$$\mu(A(\mathbf{x}, t)) \leq \alpha(t), \quad \beta(t) \leq \mu\left(-A(\mathbf{x}, t)\right) \quad \forall t \geq 0, \forall \mathbf{x} \in R^n \tag{2.77}$$

Then:

$$\|\mathbf{x}(t_0)\|\exp\left(\int_{t_0}^{t} -\beta(\tau)\mathrm{d}\tau\right) \leq \|\mathbf{x}(t)\| \leq \|\mathbf{x}(t_0)\|\exp\left(\int_{t_0}^{t} \alpha(\tau)\mathrm{d}\tau\right) \tag{2.78}$$

Proof:

We first show (2.76). We have from (2.75):

$$\begin{aligned}
\mathbf{x}(t+\delta t) &= \mathbf{x}(t) + A(\mathbf{x}, t)\mathbf{x}(t)\delta t + o(\delta t) && (2.79)\\
&= \left[I + \delta t A(\mathbf{x}, t)\right]\mathbf{x}(t) + o(\delta t) && (2.80)\\
\Rightarrow \|\mathbf{x}(t+\delta t)\| &\leq \|\left[I + \delta t A(\mathbf{x}, t)\right]\mathbf{x}(t)\| + \|o(\delta t)\| && (2.81)\\
&\leq \|I + \delta t A(\mathbf{x}, t)\|_i\|\mathbf{x}(t)\| + \|o(\delta t)\| && (2.82)
\end{aligned}$$

$$\Rightarrow \|\mathbf{x}(t+\delta t)\| - \|\mathbf{x}(t)\| \leq \{\|I + \delta t A(\mathbf{x},t)\|_i - 1\} \|\mathbf{x}(t)\| + \|o(\delta t)\| \quad (2.83)$$

$$\Rightarrow \frac{d^+}{\mathrm{dt}}\|\mathbf{x}(t)\| \leq \lim_{\delta t \to 0^+} \left\{ \frac{\|I + \delta t A(\mathbf{x},t)\|_i - 1}{\delta t} \right\} \|\mathbf{x}(t)\| \quad (2.84)$$

Equation (2.78) then follows by multiplying both sides of the inequality:

$$\frac{d^+}{\mathrm{dt}}\|\mathbf{x}(t)\| \leq \lim_{\delta t \to 0^+} \left\{ \frac{\|I + \delta t A(\mathbf{x},t)\|_i - 1}{\delta t} \right\} \|\mathbf{x}(t)\| \quad (2.85)$$

$$\leq \alpha(t)\|\mathbf{x}(t)\| \quad (2.86)$$

with the integrating factor:

$$\exp\left(-\int_{t_0}^{t} \alpha(\tau)\mathrm{d}\tau\right) \quad (2.87)$$

Q.E.D.

The matrix measures associated with the $1, 2$ and ∞ norms are given by:

$$\|\mathbf{x}\|_\infty \equiv \max_i(x_i) \rightarrow \mu_\infty(A) = \max_i \left(a_{ii} + \sum_{j \neq i} |a_{ij}| \right) \quad (2.88)$$

$$\|\mathbf{x}\|_1 \equiv \sum_{i=1}^{n} |x_i| \rightarrow \mu_1(A) = \max_j \left(a_{jj} + \sum_{i \neq j} |a_{ij}| \right) \quad (2.89)$$

$$\|\mathbf{x}\|_2 \equiv \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2} \rightarrow \mu_2(A) = \lambda_{max}\left(A^T + A\right)/2 \quad (2.90)$$

The usefulness of matrix measures becomes apparent by examining equations (2.77) and (2.78) and noting that $\mu(A)$ can have *negative* values and thus be used e.g. to show that $\|\mathbf{x}\| \rightarrow \mathbf{0}$. The following example shows how this can be done:

Example 2.3.1 Consider the following set of o.d.e.'s:

$$\dot{x}_1 = -(3 + x_1 x_2)^2 x_1 + \cos(x_1)\sin(x_2)x_2 \quad (2.91)$$

$$\dot{x}_2 = x_1 \sin(x_1^3) - x_2 x_1^2 - 2x_2 \quad (2.92)$$

These can be put in quasilinear form as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -(3+x_1x_2)^2 & \cos(x_1)\sin(x_2) \\ \sin(x_1^3) & -2-x_1^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{2.93}$$

For $A(\mathbf{x})$ above we have:

$$\mu_1(A(\mathbf{x})) < -1 \tag{2.94}$$

$$\mu_\infty(A(\mathbf{x})) < -1 \tag{2.95}$$

Note that the quasilinear form is non-unique (see Section 2.1), and that the realization above is much more useful than the alternative:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -(3+x_1x_2)^2 & \cos(x_1)\sin(x_2) \\ \sin(x_1^3)-x_1x_2 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{2.96}$$

where now the $-x_1x_2$ term in $A(2,1)$ can become unbounded and thus make it impossible to establish the row or column dominance of the diagonal terms as required by equations (2.88) and (2.89)

Remarks:

- From equation (2.90) we see that the system (2.75) will be stable if $A(\mathbf{x})$ is symmetric and negative definite.
- Equations (2.88) and (2.89) show that in order to have $\mu(A(\mathbf{x})) < 0$ (which we need for stability, see equation (2.78)) the diagonal terms of $A(\mathbf{x})$ have to be negative and dominate the off-diagonal terms. Unfortunately, when dealing with physical systems we often naturally obtain dynamical equations that do not result in $A(\mathbf{x})$ being diagonally dominant as we see in the following example and is also apparent in Section 2.2.

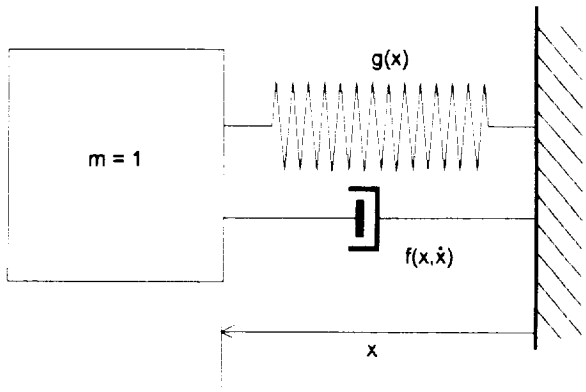


Figure 2-2: Nonlinear Spring Mass Damper

Example 2.3.2 Consider the spring mass damper system shown in Figure (2-2). The spring and damper are both nonlinear with stiffness $g(x)$ and damping characteristic $f(x, \dot{x})$ respectively. The dynamics are governed by:

$$\ddot{x} + f(x, \dot{x})\dot{x} + g(x)x = 0 \tag{2.97}$$

These equations are then conveniently put into the following quasilinear form:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -g(x) & -f(x, \dot{x}) \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \tag{2.98}$$

Here we see that for the 1 and ∞-norm cases we are stymied by the fact that $A(1,1) = 0$. Using the 2-norm does not help either because we see that:

$$\det\left(\lambda I - (A^T + A)\right) = \lambda^2 + 2f\lambda - (g-1)^2 \tag{2.99}$$

which has roots at:

$$f\left(-1 \pm \left(1 + \frac{(g-1)^2}{f^2}\right)^{1/2}\right) \tag{2.100}$$

at least one of which will be non-negative. Hence:

$$\mu(A(\mathbf{x})) \geq 0, \quad \forall \mathbf{x} \tag{2.101}$$

The challenge seems to be to find a transformation $\mathbf{z} = T(\mathbf{x})\mathbf{x}$ such that in the transformed system the matrix $A(\mathbf{x})$ will have negative entries on the diagonal that dominate the off-diagonal terms. If $A(\mathbf{x})$ is in phase variable form and its eigenvalues change relatively slowly we can use a Vandermonde matrix to effect the transform. For instance, given:

$$A(\mathbf{x}) = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \dots & 1 \\ a_0(\mathbf{x}) & a_1(\mathbf{x}) & a_2(\mathbf{x}) & \dots & a_{n-1}(\mathbf{x}) \end{bmatrix} \tag{2.102}$$

with eigenvalues $\lambda_1(\mathbf{x}), \lambda_2(\mathbf{x}), \dots \lambda_n(\mathbf{x})$. Then using the coordinate transform:

$$\mathbf{z} = T^{-1}(\mathbf{x})\mathbf{x} \tag{2.103}$$

where:

$$T(\mathbf{x}) = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ \lambda_1(\mathbf{x}) & \lambda_2(\mathbf{x}) & \lambda_3(\mathbf{x}) & \dots & \lambda_n(\mathbf{x}) \\ \lambda_1(\mathbf{x})^2 & \lambda_2(\mathbf{x})^2 & \lambda_3(\mathbf{x})^2 & \dots & \lambda_n(\mathbf{x})^2 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ \lambda_1(\mathbf{x})^{n-1} & \lambda_2(\mathbf{x})^{n-1} & \lambda_3(\mathbf{x})^{n-1} & \dots & \lambda_n(\mathbf{x})^{n-1} \end{bmatrix} \tag{2.104}$$

we get:

$$\dot{\mathbf{z}} = \Lambda(\mathbf{z})\mathbf{z} - T^{-1}\dot{T}\mathbf{z} \tag{2.105}$$

so that if the eigenvalues change relatively slowly we have $\dot{T} \approx 0$ and the transformed system will be diagonally dominant.

2.4 Summary

In this chapter we established the following:

- A broad class of dynamical systems of practical importance can be represented in quasilinear form.
- The dynamics of space-based flexible mechanical systems can easily be expressed in quasilinear form.

We also showed some *sufficient* conditions for the stability of:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} \tag{2.106}$$

based on the properties of $A(\mathbf{x})$. In particular we noted that:

- If $A(\mathbf{x})$ had negative real eigenvalues bounded away from the imaginary axis and the rate of change of the eigenvectors was small enough the system would be stable.
- If any matrix measure of $A(\mathbf{x})$, say $\mu(A(\mathbf{x}))$, is "essentially negative", i.e. $\lim_{t\to\infty} \exp\left(\int_0^t \mu(A(\mathbf{x}(\tau)))\mathrm{d}\tau\right) = 0$ the system will be stable. Such cases would for example occur when:
  - $A(\mathbf{x})$ had negative diagonal entries which dominated the off-diagonal terms.
  - $A(\mathbf{x})$ was symmetric and negative definite.

We should emphasize that these tests provide only *sufficient* conditions for stability based on the properties of $A(\mathbf{x})$ alone and that systems may well be stable even when these tests fail.

In the next chapter we will show how we exploit the quasilinear form to develop controllers for some nonlinear systems, and also investigate further the stability question using Zubov's method.

Chapter 3

Quasilinear Controllers with Zero Look-ahead

3.1 Introduction

As indicated in Chapter 1, we would like to have a controller synthesis method that

- is generic (i.e. can be applied in a straightforward fashion to most systems)
- guarantees global stability
- provides the designer with a convenient way to adjust the system's response.

We also noted that there are few methods that meet these requirements. An attractive method was that of global feedback linearization but it could only be applied to a limited class of systems. Furthermore we saw that even when systems were globally feedback linearizable, the process of actually obtaining the feedback laws could still be difficult. Another approach was to synthesize control laws by solving an optimization problem. We noted that in principle the optimization approach satisfied our

requirements for an ideal synthesis method but could be computationally demanding. In this chapter we will consider a method for control system design that is related to the optimal control approach but requires less computation. We will refer to it as the continuous Ricatti design method (see also [53] for related work).

The method we introduce exploits the quasilinear form discussed in Chapter 2. It has been successfully applied to nonlinear systems in simulation studies and partially meets our requirements for a control synthesis method, i.e. it is generic in nature and provides the designer with design parameters to conveniently influence system responses. However only local stability can be guaranteed. The stability domain for the closed loop system will have to be determined "after the fact" (see also Section 3.4).

3.2 The Continuous Ricatti Design Method

We develop the continuous Ricatti design method using the Hamilton-Jacobi-Bellman (H-J-B) theory [28] for optimal control as a starting point. By utilizing the quasilinear form we will find a candidate solution to the H-J-B-equation and an associated feedback law. We will use this associated feedback law to actually control nonlinear systems despite the fact that it may not be optimal (non-optimal because it is found from a *candidate* solution to the H-J-B equation).

3.2.1 The H-J-B-Equation and Quasilinear Dynamics

H-J-B-theory deals with the following nonlinear optimal control problem:

Find the control input $\mathbf{u}$ *that minimizes the cost function:*

$$J = \int_{t_0}^{t_f} \mathcal{L}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) \mathrm{d}\tau \tag{3.1}$$

where:

$$\mathcal{L}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) \geq 0 \tag{3.2}$$

and the system dynamics are given by:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \tag{3.3}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.4}$$

To solve the optimization problem above H-J-B theory uses the imbedding principle and considers the problem above as a specific case of a larger set of problems. The larger set of problems is constructed by allowing the initial condition to take any admissible value (instead of only $\mathbf{x}_0$) and the initial time to take any value between t_0 and t_f. Thus the cost function for the larger set of problems becomes a function of the variable initial state, say $\mathbf{x}$, and variable initial time, say t, viz:

$$J(\mathbf{x}, t) = \int_t^{t_f} \mathcal{L}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) \mathrm{d}\tau \tag{3.5}$$

Assuming that a solution to the optimization problem exists it can be shown (see e.g. [28]) that the optimal control for the original problem (3.1)-(3.4) is given by:

$$\mathbf{u}_{opt} = \arg\min_{\mathbf{u}} \left\{ (\mathcal{L}(\mathbf{x}, \mathbf{u}, \tau)) + \frac{\partial J(\mathbf{x}, t)}{\partial \mathbf{x}} (\mathbf{f}(\mathbf{x}, \mathbf{u}, t)) \right\} \tag{3.6}$$

where $J(\mathbf{x}, t)$ is found by solving the Hamilton-Jacobi-Bellman partial differential equation (H-J-B-equation):

$$-\frac{\partial J(\mathbf{x}, t)}{\partial t} = \min_{\mathbf{u}} \left\{ (\mathcal{L}(\mathbf{x}, \mathbf{u}, \tau)) + \frac{\partial J(\mathbf{x}, t)}{\partial \mathbf{x}} (\mathbf{f}(\mathbf{x}, \mathbf{u}, t)) \right\} \tag{3.7}$$

subject to the boundary conditions:

$$J(0, t) = 0 \tag{3.8}$$

$$J(\mathbf{x},t) \geq 0 \quad \forall \mathbf{x} \neq 0 \tag{3.9}$$

It can furthermore be shown that these conditions provide *both* necessary and sufficient conditions for the control to be optimal.

As a special case of the problem above consider the situation where:

- we have an infinite time interval, i.e. $t_0 = 0, t_f \to \infty$
- we have quasilinear dynamics, i.e. equation (3.3) can be put in the form

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{3.10}$$

- and the cost function is quadratic, and does not explicitly depend on time, i.e.:

$$\mathcal{L}(\mathbf{x},\mathbf{u},\tau) = \mathcal{L}(\mathbf{x},\mathbf{u}) = \frac{1}{2}\left(\mathbf{x}^T Q\mathbf{x} + \mathbf{u}^T R\mathbf{u}\right) \tag{3.11}$$

Under these conditions we get the following expression for the optimal control:

$$\mathbf{u}_{opt} = -R^{-1}B^T(\mathbf{x})\left(\frac{\partial J(\mathbf{x})}{\partial \mathbf{x}}\right)^T \tag{3.12}$$

while H-J-B-equation simplifies to:

$$0 = \mathbf{x}^T Q\mathbf{x} + 2\frac{\partial J(\mathbf{x})}{\partial \mathbf{x}}A(\mathbf{x})\mathbf{x} - \frac{\partial J(\mathbf{x})}{\partial \mathbf{x}}B(\mathbf{x})R^{-1}B(\mathbf{x})^T\frac{\partial J(\mathbf{x})}{\partial \mathbf{x}}^T \tag{3.13}$$

Remarks:

- We have set $\frac{\partial J(\mathbf{x},t)}{\partial t} = 0$ in equation (3.7) because we have an infinite time problem.
- We see that all we need to construct the optimal control is to find an expression for $\frac{\partial J(\mathbf{x})}{\partial \mathbf{x}}$; we do not explicitly require an expression for $J(\mathbf{x})$.

- It can be shown that given stabilizability and detectability assumptions the closed loop system will be stable [45].

Now assume that the infinite time, nonlinear, quadratic cost optimization problem is sufficiently well behaved for $\frac{\partial J(\mathbf{x})}{\partial \mathbf{x}}$ to be continuously differentiable (this assumption is often made in deriving the H-J-B equation). Then according to Section (2.1) there exists a matrix $\bar{P}(\mathbf{x})$ such that:

$$\left(\frac{\partial J(\mathbf{x})}{\partial \mathbf{x}}\right)^T = \bar{P}(\mathbf{x})\mathbf{x} \tag{3.14}$$

Using this representation for $\left(\frac{\partial J(\mathbf{x})}{\partial \mathbf{x}}\right)^T$ we are led to consider the equation:

$$\begin{aligned} 0 &= \mathbf{x}^T Q\mathbf{x} + 2\mathbf{x}^T \bar{P}^T(\mathbf{x})A(\mathbf{x})\mathbf{x} - \mathbf{x}^T \bar{P}^T(\mathbf{x})B(\mathbf{x})R^{-1}B^T(\mathbf{x})\bar{P}(\mathbf{x})\mathbf{x} && (3.15)\\ &= \mathbf{x}^T \left(Q + \bar{P}^T(\mathbf{x})A(\mathbf{x}) + A^T(\mathbf{x})\bar{P}(\mathbf{x}) - \bar{P}^T(\mathbf{x})B(\mathbf{x})R^{-1}B^T(\mathbf{x})\bar{P}(\mathbf{x})\right)\mathbf{x} && (3.16)\end{aligned}$$

One solution to (3.16) can be found by setting $\bar{P}(\mathbf{x}) = P(\mathbf{x})$ where $P(\mathbf{x})$ is the positive definite symmetric solution to the matrix Ricatti Equation:

$$0 = Q + P(\mathbf{x})A(\mathbf{x}) + A^T(\mathbf{x})P(\mathbf{x}) - P(\mathbf{x})B(\mathbf{x})R^{-1}B^T(\mathbf{x})P(\mathbf{x}) \tag{3.17}$$

Let us examine the solution we have obtained in more detail. Although:

$$P(\mathbf{x})\mathbf{x} \tag{3.18}$$

satisfies equation (3.16), it has to meet certain requirements before we can say that we have a solution to the H-J-B-equation (3.13) (see also [45] for conditions that guarantee that a control is optimal). The requirements are:

(1) $P(\mathbf{x})\mathbf{x}$ must be the gradient of some scalar function. That is, there must be

some scalar function $V(\mathbf{x})$ such that:

$$\left(\frac{\partial V}{\partial \mathbf{x}}\right)^T = P(\mathbf{x})\mathbf{x} \tag{3.19}$$

At this point we have no guarantee that there exists such a scalar function.

(2) If it is true that $P(\mathbf{x})\mathbf{x}$ is the gradient of some scalar function, then $V(\mathbf{x})$ has to furthermore satisfy the conditions (3.8),(3.9).

The following lemma shows that if that P is the positive definite solution to (3.17) the second condition will be met.

Lemma 3.2.1 *If*

$$\left(\frac{\partial V}{\partial \mathbf{x}}\right)^T = P(\mathbf{x})\mathbf{x} \tag{3.20}$$

where $P(\mathbf{x})$ is a positive definite matrix for all $\mathbf{x}$ and $V(\mathbf{0}) = 0$ then we will have:

$$V(\mathbf{x}) > 0 \quad \forall \mathbf{x} \neq \mathbf{0} \tag{3.21}$$

Proof:

Fix $\mathbf{x}$ and consider $V(\mu\mathbf{x})$ as a function of the scalar variable μ. Then

$$\begin{aligned} V(\mathbf{x}) &= \int_0^1 \frac{\mathrm{d}}{\mathrm{d}\mu} V(\mu\mathbf{x})\mathrm{d}\mu && (3.22)\\ &= \int_0^1 \left(\frac{\partial V}{\partial \mathbf{x}}\right)\Big|_{\mu\mathbf{x}} \mathbf{x}\,\mathrm{d}\mu && (3.23)\\ &= \int_0^1 \left(\mu\mathbf{x}^T P(\mu\mathbf{x})\right)\mathbf{x}\,\mathrm{d}\mu && (3.24)\\ &\geq \int_0^1 \mu\mathbf{x}^T\mathbf{x}\,\lambda_{\min}\left[P(\mu\mathbf{x})\right]\,\mathrm{d}\mu && (3.25) \end{aligned}$$

where $\lambda_{\min}\left[P(\mu\mathbf{x})\right]$ denotes the minimum eigenvalue of $P(\mu\mathbf{x})$. Since $P(\mathbf{x})$ is positive definite for all $\mathbf{x}$ we have $\lambda_{\min}\left[P(\mu\mathbf{x})\right] > 0$, and it follows that $V(\mathbf{x}) > 0$ for all $\mathbf{x} \neq 0$.

Q.E.D.

With Lemma 3.2.1 in hand we can thus conclude that

$$P(\mathbf{x})\mathbf{x} \tag{3.26}$$

with $P(\mathbf{x})$ the positive definite solution to equation (3.17), will be a solution to the H-J-B equation for those cases where $P(\mathbf{x})\mathbf{x}$ actually is the gradient of some scalar function. (Note that when we have a linear system, P does not depend on the state and thus $P(\mathbf{x})\mathbf{x} = P\mathbf{x}$ will satisfy the gradient condition).

3.2.2 The Continuous Ricatti Design Feedback Law

For the continuous Ricatti design method we assume that the nonlinear system has been put in the quasilinear form of equation (3.10). We then use the *form* of the solution shown in the previous section without regard to whether condition (3.19) is satisfied or not. Specifically we use the following state feedback law:

$$\mathbf{u} = -K(\mathbf{x})\mathbf{x} \tag{3.27}$$

where

$$K(\mathbf{x}) = R^{-1}B^T(\mathbf{x})P(\mathbf{x}) \tag{3.28}$$

and $P(\mathbf{x})$ is the symmetric positive semi-definite matrix solution to the matrix Ricatti equation:

$$0 = Q + P(\mathbf{x})A(\mathbf{x}) + A^T(\mathbf{x})P(\mathbf{x}) - P(\mathbf{x})B(\mathbf{x})R^{-1}B^T(\mathbf{x})P(\mathbf{x}) \tag{3.29}$$

Remarks:

- The method we propose is generic in nature since a large class of nonlinear systems can be put in quasilinear form (see Section 2.1).

- We have found that when using the feedback law (3.27) the responses of the closed loop system could be influenced in a predictable manner by our choice of Q and R.

- The feedback law is based on the dynamics of the system "frozen" at the current state. It "ignores" future behavior of $A(\mathbf{x})$ and $B(\mathbf{x})$, hence we refer to it as a control with *zero look-ahead* (see Chapter 4 for more discussion on the look-ahead concept).

- Because condition (3.19) may not be satisfied we cannot claim optimality nor global stability for the closed loop system. However we have found that continuous Ricatti design controllers generally are better behaved than designs based on the linearized dynamics of the systems (see also Section 3.3.3 for further discussion of the stability properties).

Example 3.2.1 Consider the nonlinear system:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \tag{3.30}$$

$$= \begin{bmatrix} 0.5\sin(x_2)x_1 + x_2 \\ x_1 - x_1 x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u \tag{3.31}$$

One approach to developing a controller for this system would be to find a feedback law based on the linearized dynamics of the system. For example, linearizing (3.31) at the equilibrium point $\mathbf{x} = \mathbf{0}$, gives:

$$\dot{\delta\mathbf{x}} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{0}} \delta\mathbf{x} + \left.\frac{\partial(\mathbf{g}u)}{\partial u}\right|_{\mathbf{x}=\mathbf{0}} u \tag{3.32}$$

$$= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \delta x_1 \\ \delta x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u \tag{3.33}$$

A Linear Quadratic Regulator feedback law based on the linearized dynamics and cost function:

$$J = \int_0^\infty \left(\delta\mathbf{x}^T Q \delta\mathbf{x} + uRu\right) \mathrm{dt} \tag{3.34}$$

where:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad R = 1 \tag{3.35}$$

gives the control law:

$$u = -K_{lin}\mathbf{x} \tag{3.36}$$

where:

$$K_{lin} = \begin{bmatrix} 1.366 & 1.366 \end{bmatrix} \tag{3.37}$$

If we apply this constant gain feedback to the full nonlinear system we get closed loop responses as shown in the phase plane plot of Figure 3-1. The closed loop system is locally stable (as it should be because its linearization is stable — see also Section 3.3.1) — but eventually becomes unstable.

However if we apply the continuous Ricatti design method to the same system using the same Q and R, and the following quasilinear form for the dynamics of equation (3.31):

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0.5\sin(x_2) & 1 \\ 1 & -x_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u \tag{3.38}$$

we get the "non-local" stable behavior shown in the phase plane plot of Figure 3-2. We also see that for the continuous Ricatti design method, the trajectories starting in the quadrant where $x_1 < 0, x_2 < 0$ take a more direct route to the origin.

3.2.3 Controllers for Combined Tracking and Regulation

In this section we develop methods to achieve our second goal for the nonlinear design methodology, i.e. to find design variables that we can use in the synthesis process to adjust system responses. Our approach is to construct an appropriate quadratic regulator problem and then use the feedback law (3.27) given by the candidate solution to the H-J-B equation discussed in the previous section. We have found that despite the fact that the candidate solution may be non-optimal/unstable we could

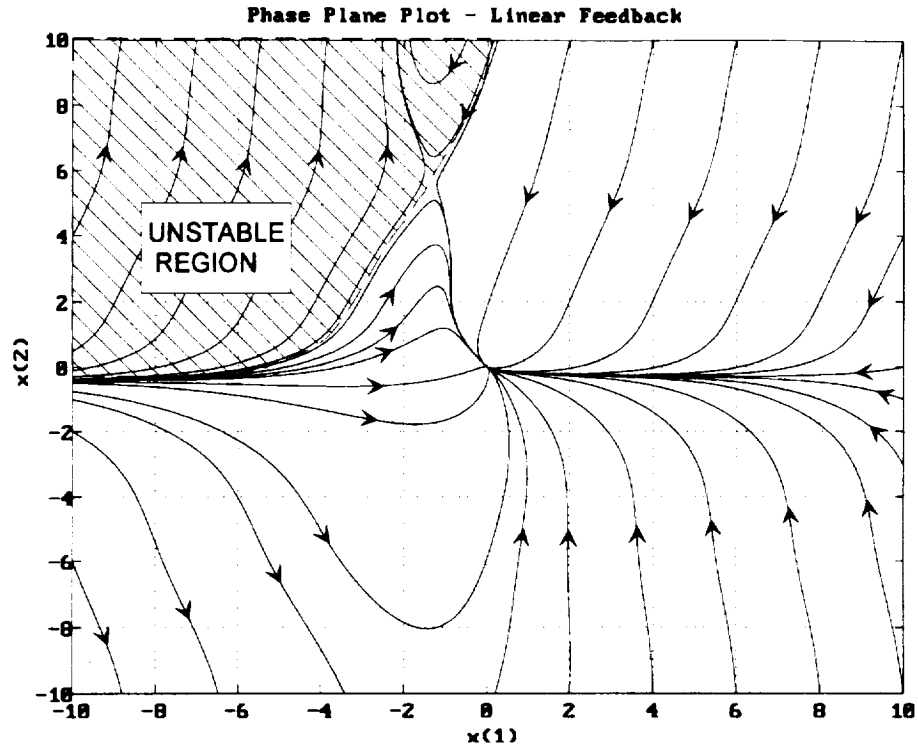


Figure 3-1: Phase Plane Plot for Linear Feedback

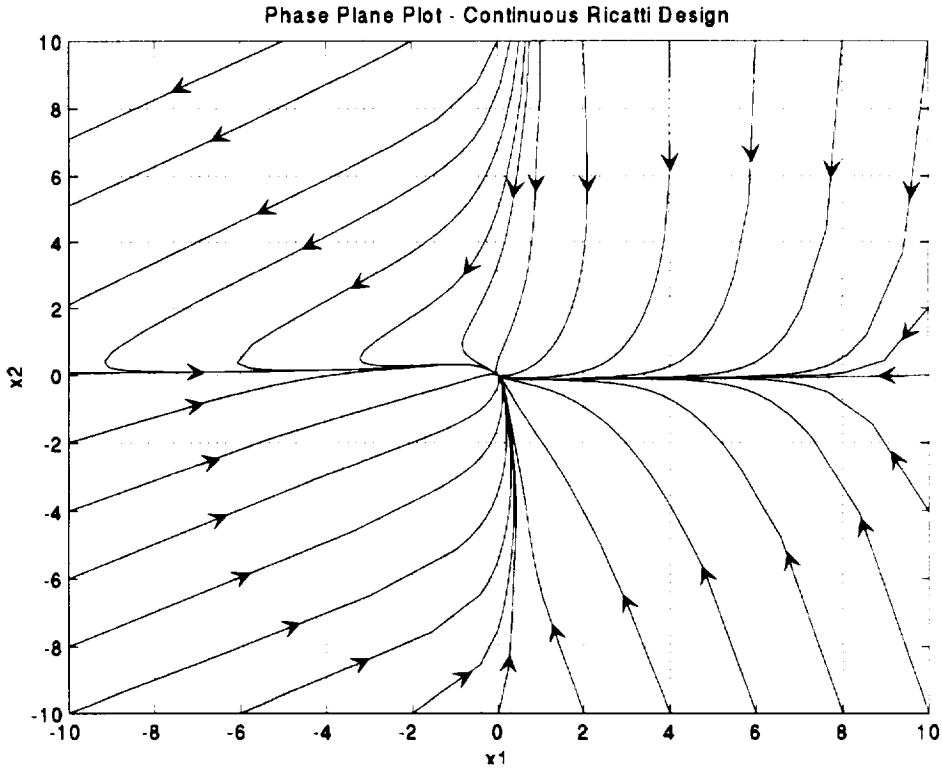


Figure 3-2: Phase Plane Plot for Continuous Ricatti Design

still modify system behavior in a predictable way by adjusting the cost function (the Q and R matrices in equation (3.11)).

Our goal is to find a control law that causes output variables, say $\mathbf{y} = C\mathbf{x}$, to track desired values, say $\mathbf{y}_d$, and at the same time cause other state variables, say $\boldsymbol{\eta}$, to remain small. For example, we can think of $\mathbf{y}$ as variables that describe the gross motion of a flexible manipulator and $\boldsymbol{\eta}$ as its flex mode deflections. Of course depending on the system we are dealing with, the physics of the problem may not allow us to simultaneously achieve both accurate tracking ($\mathbf{y} \approx \mathbf{y}_d$) as well as good regulation of the $\boldsymbol{\eta}$ variables. However by using a cost function approach, we have been able to conveniently make the trade-off between tracking accuracy and amount of regulation.

The generality of this combined tracking and regulating goal becomes clearer if we recall (see [59] and Section 1.1.2) that through partial feedback linearization we can generally transform multivariable non-linear systems of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{B}(\mathbf{x})\mathbf{u} \tag{3.39}$$

$$= A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{3.40}$$

$$\mathbf{y} = C\mathbf{x} \tag{3.41}$$

to the following form:

$$\begin{bmatrix} y_1^{(r_1)} \\ y_2^{(r_2)} \\ \vdots \\ y_m^{(r_m)} \end{bmatrix} = \begin{bmatrix} \alpha_1(\mathbf{x}) \\ \alpha_2(\mathbf{x}) \\ \vdots \\ \alpha_m(\mathbf{x}) \end{bmatrix} + \begin{bmatrix} \boldsymbol{\beta}_1^T(\mathbf{x})\mathbf{u} \\ \boldsymbol{\beta}_2^T(\mathbf{x})\mathbf{u} \\ \vdots \\ \boldsymbol{\beta}_m^T(\mathbf{x})\mathbf{u} \end{bmatrix} \tag{3.42}$$

$$\stackrel{\text{def}}{=} \begin{bmatrix} \alpha_1(\mathbf{x}) \\ \alpha_2(\mathbf{x}) \\ \vdots \\ \alpha_m(\mathbf{x}) \end{bmatrix} + E(\mathbf{x})\mathbf{u} \tag{3.43}$$

$$\dot{\eta} = \mathbf{w}(\eta, \mathbf{x}, \mathbf{u}) \tag{3.44}$$

where $y_i^{(r)}$ denotes the r'th derivative of y_i. Here we have a generic separation of our state variables into two classes viz.

- $\mathbf{y} = [y_1, \ldots y_m]^T$ which are output variables which must track the desired trajectories $\mathbf{y}_d(t)$
- η which are other state variables which we may want to keep small.

Note that the continuous Ricatti design method does *not* require us to transform our dynamics to the partially feedback linearized form shown here — the analysis above only serves to show that the optimization problem is generic in some sense.

Let us now construct the optimization problem for our quasilinear system. We will structure the problem in such a way that we remain with a regulator type problem (as opposed to an explicit tracking type problem [28]). Therefore we assume that the variables $\mathbf{y}_d$ are the outputs of some reference model, that will generally be a stable linear system driven by an external input, viz:

$$\dot{\mathbf{x}}_d = A_d\mathbf{x}_d + B_d\mathbf{r} \tag{3.45}$$

$$\mathbf{y}_d = C_d\mathbf{x}_d \tag{3.46}$$

where:

$\mathbf{x}_d$ is the l dimensional state vector of the reference model.

$\mathbf{r}$ is an external input driving the reference model.

We can now augment our basic plant dynamics (3.40) with our reference model to form the system:

$$\dot{\mathbf{z}} = F(\mathbf{z})\mathbf{z} + G(\mathbf{z})\mathbf{u} + \Gamma\mathbf{r} \tag{3.47}$$

$$\mathbf{e} = \begin{bmatrix} C & -C_d \end{bmatrix}\mathbf{z} \tag{3.48}$$

where:

$$\mathbf{z} \stackrel{\text{def}}{=} \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_d \end{bmatrix} \quad \mathbf{e} \stackrel{\text{def}}{=} \mathbf{y} - \mathbf{y}_d \tag{3.49}$$

and

$$F(\mathbf{z}) = \begin{bmatrix} A(\mathbf{x}) & 0 \\ 0 & A_d \end{bmatrix} \quad G(\mathbf{z}) = \begin{bmatrix} B(\mathbf{x}) \\ 0 \end{bmatrix} \quad \Gamma = \begin{bmatrix} 0 \\ B_d \end{bmatrix} \tag{3.50}$$

In order to define a regulator problem we will temporarily set $\mathbf{r} = \mathbf{0}$ but will show in Lemma 3.2.2 that under certain conditions we can get $\mathbf{y}$ to track $\mathbf{y}_d$ even when $\mathbf{r} \neq \mathbf{0}$.

The cost function for our problem will penalize: the tracking error $\mathbf{e} \stackrel{\text{def}}{=} \mathbf{y} - \mathbf{y}_d$, excursions of those states which we want to regulate, as well as the control input, viz:

$$\begin{aligned} J &= \int_0^\infty \left(\mathbf{e}^T Q_y \mathbf{e} + \mu_x \mathbf{x}^T Q_x \mathbf{x} + \rho^2 \mathbf{u}^T \mathbf{u} \right) \mathrm{dt} && (3.51) \\ &= \int_0^\infty \left((\mathbf{y} - \mathbf{y}_d)^T Q_y (\mathbf{y} - \mathbf{y}_d) + \mu_x \mathbf{x}^T Q_x \mathbf{x} + \rho^2 \mathbf{u}^T \mathbf{u} \right) \mathrm{dt} && (3.52) \\ &\stackrel{\text{def}}{=} \int_0^\infty \left(\mathbf{z}^T Q \mathbf{z} + \mathbf{u}^T R \mathbf{u} \right) \mathrm{dt} && (3.53) \end{aligned}$$

where:

$$Q \stackrel{\text{def}}{=} \begin{bmatrix} C^T Q_y C + \mu_x Q_x & -C^T Q_y C_d \\ -C_d^T Q_y C & C_d^T Q_y C_d \end{bmatrix} \tag{3.54}$$

With the cost function as defined above (and $\mathbf{r} = 0$) we have a problem in a format to which we can apply the continuous Ricatti design method as discussed in Section 3.2.2. We will furthermore show that independent of the question whether the continuous Ricatti design method is optimal or not we can get $\mathbf{y}$ to track $\mathbf{y}_d$ under certain conditions. To do this we need the following result by Kwakernaak and Sivan [33]:

Theorem 3.2.1 *Consider the time-invariant stabilizable and detectable linear system:*

$$\begin{aligned} \dot{\mathbf{z}} &= F_f \mathbf{z} + G_f \mathbf{u} && (3.55) \\ \mathbf{y} &= C_f \mathbf{z} && (3.56) \end{aligned}$$

where G_f and C_f are assumed to have full rank. Consider also the cost function:

$$\int_0^\infty \left(\mathbf{z}^T C_f^T Q C_f \mathbf{z} + \mathbf{u}^T R \mathbf{u}\right) \mathrm{dt} \tag{3.57}$$

with Q, R positive-definite and symmetric. Let

$$R = \rho^2 N \tag{3.58}$$

with N positive definite and ρ a positive scalar. Let $\bar{P}_\rho$ be the steady-state solution of the Ricatti-equation:

$$-\dot{P}_\rho(t) = C_f^T Q C_f + P_\rho(t) F_f + F_f^T P_\rho(t) - P_\rho(t) G_f R^{-1} G_f^T P_\rho(t) \tag{3.59}$$

$$P_\rho(t_f) = 0 \tag{3.60}$$

Then the following facts hold:

1. *The limit:*

$$\lim_{\rho \to 0} \bar{P}_\rho = P_0 \tag{3.61}$$

   exists.

2. *Let $\mathbf{z}_\rho(t)$,$t \geq 0$, denote the response of the controlled variable for the regulator that is steady-state optimal for $R = \rho^2 N$. Then*

$$\lim_{\rho \to 0} \int_0^\infty \mathbf{z}_\rho^T Q \mathbf{z}_\rho \, \mathrm{dt} = \mathbf{z}^T(0) P_0 \mathbf{z}(0) \tag{3.62}$$

   (i.e. the term $\int_0^\infty \mathbf{u}^T N \mathbf{u} \mathrm{dt}$ remains finite as $\rho \to 0$)

3. *If $\dim(\mathbf{y}) > \dim(\mathbf{u})$ then $P_0 \neq 0$*

4. *If $\dim(\mathbf{y}) = \dim(\mathbf{u})$ and the numerator polynomial $\Psi(s)$ of the open-loop transfer matrix $H(s) = C_f(sI - F_f)^{-1} G_f$ is nonzero, $P_0 = 0$ if and only if all the zeros of the numerator polynomial [33] have non-positive real parts.*

5. *If* $\dim(\mathbf{y}) < \dim(\mathbf{u})$, *then* $P_0 = 0$ *if there exists a rectangular matrix* M *such that the numerator polynomial* $\bar{\Psi}$ *of the square transfer matrix* $C_f(sI - A)^{-1}G_f M$ *is non-zero and has zeros with non-positive real parts only.*

6. *If* $\dim(\mathbf{y}) = \dim(\mathbf{u})$ *and* $P_0 = 0$ *then:*

$$N^{\frac{1}{2}} \left(\lim_{\rho \to 0} G_f^T \frac{\bar{P}}{\rho} \right) = W Q^{\frac{1}{2}} C_f \tag{3.63}$$

where W *is some unitary matrix.*

We can now show when we can get $\mathbf{y} \to \mathbf{y}_d$ independent of the question of optimality (see also [53] for related results in a Loop Transfer Recovery setting).

Theorem 3.2.2 *For the system (3.47),(3.48) and cost-function (3.53) assume that:*

- $G(\mathbf{z}), [C \ -C_d]$ *are full rank for all* $\mathbf{z}$ *with* $\operatorname{rank}(G) = \operatorname{rank}([C \ -C_d])$
- $Q_y = I, R = \rho^2 I$ *and* ρ *is positive*
- $\{F(\mathbf{z}), G(\mathbf{z})\}$ *is stabilizable for all* $\mathbf{z}$
- $\{[C - C_d], F(\mathbf{z})\}$ *is detectable for all* $\mathbf{z}$
- *The transfer function* $[C - C_d](sI - F(\mathbf{z}))^{-1}G(\mathbf{z})$ *is minimum-phase for all* $\mathbf{z}$

If we apply the control:

$$\mathbf{u} = -K(\mathbf{z})\mathbf{z} \tag{3.64}$$

where:

$$K(\mathbf{z}) = \frac{1}{\rho^2} G(\mathbf{z})^T P(\mathbf{z}) \tag{3.65}$$

and $P(\mathbf{z})$ *is the unique positive semi-definite solution to:*

$$0 = Q + P(\mathbf{z})F(\mathbf{z}) + F^T(\mathbf{z})P(\mathbf{z}) - \frac{1}{\rho^2} P(\mathbf{z})G(\mathbf{z})G^T(\mathbf{z})P(\mathbf{z}) \tag{3.66}$$

with Q as in equation (3.54) then if $\mu_x = 0$ we will get

$$\mathbf{y} \rightarrow \mathbf{y}_d \tag{3.67}$$

as $\rho \rightarrow 0$.

Proof:

If we apply the control (3.64) to the system (3.47) we get for the closed loop dynamics:

$$\dot{\mathbf{z}} = \left[F(\mathbf{z}) - \frac{1}{\rho^2} G(\mathbf{z}) G^T(\mathbf{z}) P(\mathbf{z})\right] \mathbf{z} + \Gamma \mathbf{r} \tag{3.68}$$

$$\Rightarrow \rho \dot{\mathbf{z}} = \rho F(\mathbf{z})\mathbf{z} - G(\mathbf{z}) \left(\frac{1}{\rho} G^T(\mathbf{z}) P(\mathbf{z})\right) \mathbf{z} + \rho \Gamma \mathbf{r} \tag{3.69}$$

Now taking the limit as $\rho \rightarrow 0$ we get:

$$0 \leftarrow -G(\mathbf{z}) \left(\frac{1}{\rho} G^T(\mathbf{z}) P(\mathbf{z})\right) \mathbf{z} \tag{3.70}$$

By using Theorem (3.2.1) we get that as $\rho \rightarrow 0$:

$$0 \leftarrow -G(\mathbf{z}) W(\mathbf{z}) \left[\ C - C_d\ \right] \mathbf{z} \tag{3.71}$$

where $W(\mathbf{z})$ is a unitary matrix. Premultiplying by $G^T(\mathbf{z})$ we get:

$$0 \leftarrow G^T(\mathbf{z}) G(\mathbf{z}) W(\mathbf{z}) \left[\ C - C_d\ \right] \mathbf{z} \tag{3.72}$$

and using the facts that $G^T G$ is non-singular because G is full rank, and W is unitary we get:

$$0 \leftarrow C\mathbf{z} - C_d \mathbf{z} = \mathbf{y} - \mathbf{y}_d \tag{3.73}$$

Q.E.D.

Remarks:

- The above theorem shows that we asymptotically get perfect tracking when $\mu_x = 0$ and $\rho \to 0$. By increasing μ_x we have been able to trade off tracking accuracy relative to the amount of regulation on the state variables which we want to keep small.

- Although we may get perfect tracking with $\mu_x = 0$ we are still not guaranteed stability as can easily be seen by examining equations (3.43) and (3.44). For instance although $\mathbf{y}$ may be well behaved $\boldsymbol{\eta}$ may be unstable.

- To obtain perfect tracking through partial feedback linearization one typically requires that $E(\mathbf{x})$ in equation (3.43) be invertible. However we see that by using the continuous Ricatti Design method we do not need to perform the partial feedback linearization step and thus do not require that $E(\mathbf{x})$ be invertible.

- We have seen that we asymptotically achieve perfect tracking as $\rho \to 0$. This may give rise to the concern that we will get large control inputs. Fortunately we note that the control input requirements are essentially determined by the response/bandwidth of the reference model. Hence we can ensure that we have reasonable control inputs by suitably limiting the bandwidth of our reference model.

- If we want proportional plus integral control we can further augment our system with states which are the integral of the tracking error. This will result in the augmented system

$$\dot{\mathbf{z}} = F(\mathbf{z})\mathbf{z} + G(\mathbf{z})\mathbf{u} + \Gamma\mathbf{r} \tag{3.74}$$

where now:

$$\mathbf{z} = [\mathbf{x}, \mathbf{x}_d, \mathbf{e}_I]^T \tag{3.75}$$

$$\dot{\mathbf{e}}_I = \mathbf{y} - \mathbf{y}_d \tag{3.76}$$

$$F(\mathbf{z}) = \begin{bmatrix} A(\mathbf{x}) & 0 & 0 \\ 0 & A_d & 0 \\ C & -C_d & 0 \end{bmatrix} \quad G(\mathbf{z}) = \begin{bmatrix} B(\mathbf{x}) \\ 0 \\ 0 \end{bmatrix} \quad \Gamma = \begin{bmatrix} 0 \\ B_d \\ 0 \end{bmatrix} \tag{3.77}$$

To see how this results in a system with P-I-control let us examine the resulting control law. The closed loop control will be of the form:

$$\mathbf{u} = -K(\mathbf{z})\mathbf{z} \tag{3.78}$$

$$\stackrel{\text{def}}{=} -\begin{bmatrix} K_x & K_d & K_I \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_d \\ \mathbf{e}_I \end{bmatrix} \tag{3.79}$$

(where we have suppressed the dependence on $\mathbf{z}$ of K_x, K_d, K_I for notational convenience). If we assume that $[C - C_d]$ is full rank we can find an invertible matrix T:

$$T \stackrel{\text{def}}{=} \begin{bmatrix} C & -C_d \\ T_{21} & T_{22} \end{bmatrix} \tag{3.80}$$

which we can use to transform to a new set of state variables, viz:

$$\begin{bmatrix} \mathbf{e} \\ \mathbf{x}_r \end{bmatrix} = T \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_d \end{bmatrix} \tag{3.81}$$

We can now rewrite the control law in a form that exhibits the proportional and integral feedback terms, as follows:

$$\mathbf{u} = -\begin{bmatrix} K_x & K_d \end{bmatrix} T^{-1} T \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_d \end{bmatrix} - K_I \mathbf{e}_I \tag{3.82}$$

$$= -\begin{bmatrix} K_x & K_d \end{bmatrix} T^{-1} \begin{bmatrix} \mathbf{e} \\ \mathbf{x}_r \end{bmatrix} - K_I \mathbf{e}_I \tag{3.83}$$

$$\stackrel{\text{def}}{=} -K_{prop}\mathbf{e} - K_{int}\mathbf{e}_I - K_r \mathbf{x}_r \tag{3.84}$$

Figure 3-3 gives a block-diagram of this control law. (Note that for the quasi-linear system K_{prop}, K_{int}, K_r all depend on $\mathbf{z}$ although we have not explicitly shown this dependence).

- The approach we used in this section also provides a useful solution to the

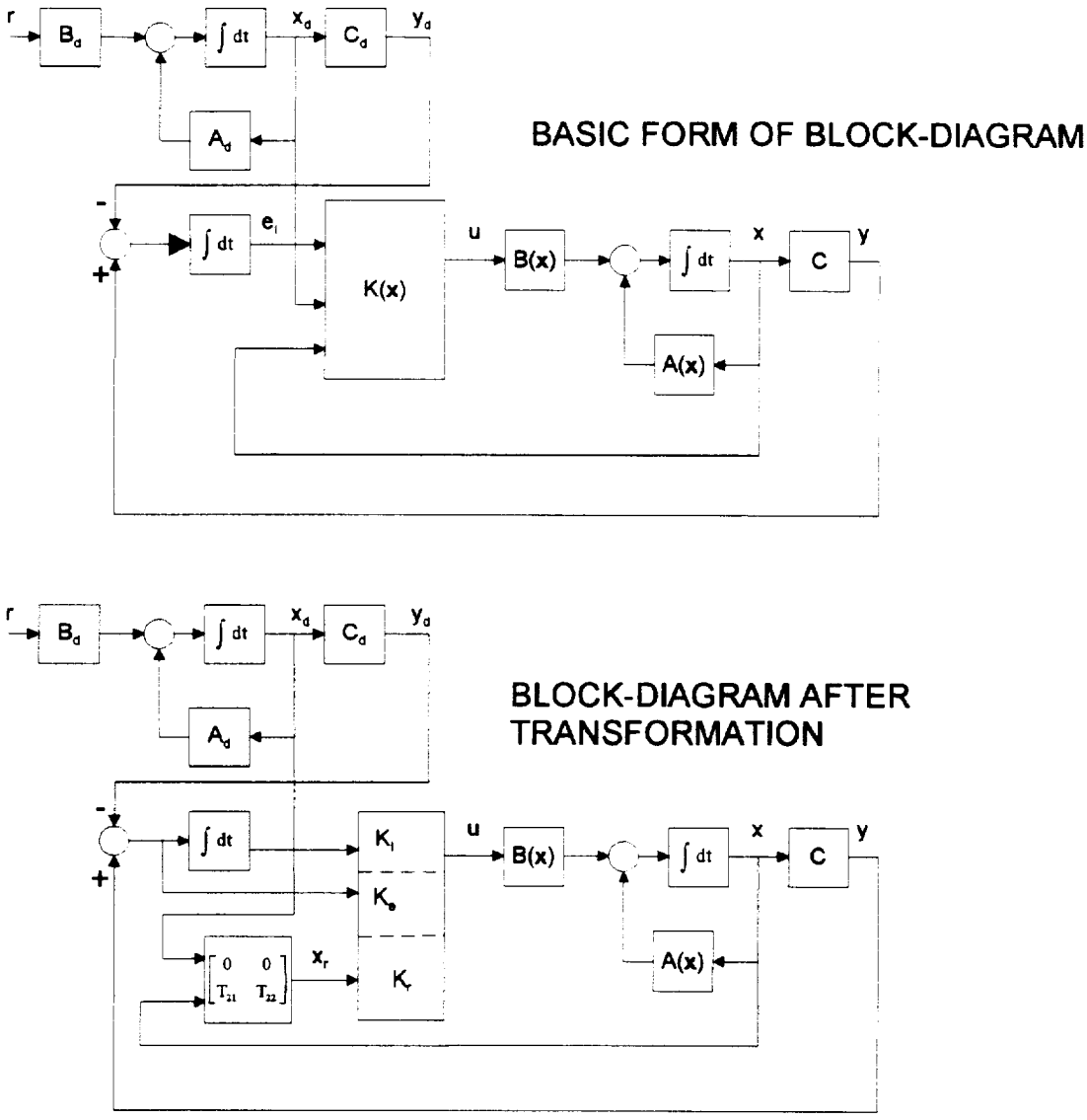


Figure 3-3: Block Diagram of Augmented System with Integral Control

so-called *LQ-Servo* problem [4] for linear time invariant systems.

3.3 Stability Analysis

In this section we examine the stability properties of the continuous Ricatti design method. It has been our experience that the method results in closed loop systems with good stability properties if we appropriately adjust the cost function. Our

approach will be to motivate our expectation of stability through analytical means but accept that we will have to assess the actual stability of the closed loop systems through computational methods. The computational methods will be discussed in Section 3.4, while we consider the following issues in this Section:

- Local stability
- Existence of multiple equilibrium points
- "Non-local" stability

We will prove that the control law will result in closed loop systems which are locally stable in the sense of Lyapunov (i.s.L.) and that the closed loop system will generally have a single equilibrium point at the origin. By using results from linear time varying systems theory we will furthermore show that the stability region can generally be expected to be large, but will not prove global asymptotic stability.

We will use the following stability definitions in our analysis [59],[63]:

Definition 3.3.1 *Assume that the system:*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{3.85}$$

has an equilibrium point at $\mathbf{x} = \mathbf{0}$, *i.e.* $\mathbf{f}(\mathbf{0}) = \mathbf{0}$. *Then*

- *The equilibrium* $\mathbf{0}$ *is* **stable i.s.L.** *if for each* $\epsilon > 0$ *there exists a* $\delta = \delta(\epsilon) > 0$ *such that*

  $$\|\mathbf{x}(0)\| < \delta(\epsilon) \Rightarrow \|\mathbf{x}(t)\| < \epsilon, \forall t \geq 0 \tag{3.86}$$

  Otherwise the equilibrium point $\mathbf{0}$ *is* **unstable**

- *The equilibrium* $\mathbf{0}$ *is* **asymptotically stable i.s.L.** *if it is stable i.s.L. and if in addition there exists some* $\eta > 0$ *such that*

  $$\|\mathbf{x}(0)\| < \eta \Rightarrow \mathbf{x}(t) \rightarrow \mathbf{0} \quad \text{as} \quad t \rightarrow \infty \tag{3.87}$$

- *The equilibrium* $\mathbf{0}$ *is* **exponentially stable** *if there exist constants* $r, \alpha, \lambda > 0$ *such that*

$$\|\mathbf{x}(t)\| \leq \alpha \|\mathbf{x}(0)\| e^{-\lambda t}, \quad \forall t > 0, \forall \|\mathbf{x}(0)\| \leq r \tag{3.88}$$

- *If the equilibrium* $\mathbf{0}$ *is stable i.s.L., or asymptotically stable, or exponentially stable for all initial states, the equilibrium point is said to be globally stable i.s.L., or globally asymptotically stable, or globally exponentially stable respectively.*

Note that exponential stability is the strongest form of stability since it implies asymptotic stability as well as stability i.s.L.

3.3.1 Local Stability Properties

As a first step in analyzing the stability properties of the continuous Ricatti design method we prove that the method yields closed loop systems that are locally stable.

To show local stability for the continuous Ricatti design method we use Lyapunov's linearization method which is based on the following theorem [59]:

Theorem 3.3.1 *Consider the system:*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{3.89}$$

$$\mathbf{f}(\mathbf{0}) = \mathbf{0} \tag{3.90}$$

Suppose that $\mathbf{f}(\mathbf{x})$ *is continuously differentiable and let* $J(\mathbf{0})$ *be the Jacobian of* $\mathbf{f}(\mathbf{x})$ *evaluated at* $\mathbf{x} = \mathbf{0}$, *i.e.:*

$$J(\mathbf{0}) \stackrel{\text{def}}{=} \left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right]\Bigg|_{\mathbf{x}=\mathbf{0}} \tag{3.91}$$

Under these conditions:

- *If all eigenvalues of* $J(\mathbf{0})$ *are strictly in the left half complex plane then* $\mathbf{0}$ *is an exponentially stable equilibrium point of (3.90).*

- *If one or more eigenvalues of $J(\mathbf{0})$ is strictly in the right-half complex plane then $\mathbf{0}$ is an unstable equilibrium point.*
- *If all the eigenvalues of $J(\mathbf{0})$ are in the left-hand complex plane with at least one eigenvalue on the $j\omega$-axis then one cannot conclude stability nor instability of the nonlinear system (3.90) from its linearized approximation.*

Theorem (3.3.1) makes it clear that in order to establish local stability of the continuous Ricatti design method it is sufficient to examine the linearized approximation of the closed loop system. With this in mind we next show how we can easily obtain the linearized approximation of a nonlinear system if we have its dynamics in quasilinear form. (Note that the proof below holds for any quasilinear realization of $\mathbf{f}(\mathbf{x})$, not only those obtained via equation (2.6)).

Lemma 3.3.1 *Consider the nonlinear system*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{3.92}$$

$$= A(\mathbf{x})\mathbf{x} \tag{3.93}$$

with $\mathbf{f}(\mathbf{x})$ continuously differentiable in an open region around $\mathbf{0}$. Then the Jacobian of $\mathbf{f}(\mathbf{x})$ is given by:

$$\left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right]\Bigg|_{\mathbf{x}=\mathbf{0}} = A(\mathbf{0}) \tag{3.94}$$

Proof:

Recall (see [46]) that the Jacobian of $\mathbf{f}(\mathbf{x})$ evaluated at $\mathbf{x}_0$ is equal to the matrix $J(\mathbf{x}_0)$ such that:

$$\frac{\mathbf{f}(\mathbf{x}_0+\mathbf{h}) - \mathbf{f}(\mathbf{x}_0) - J(\mathbf{x}_0)\mathbf{h}}{\|\mathbf{h}\|} \to 0 \quad \text{as} \quad \mathbf{h} \to \mathbf{0} \tag{3.95}$$

It can furthermore be shown that the matrix $J(\mathbf{x}_0)$ is unique [46]. For our case we want the Jacobian of:

$$\mathbf{f}(\mathbf{x}) = A(\mathbf{x})\mathbf{x} \tag{3.96}$$

evaluated at $\mathbf{x} = \mathbf{0}$. Now note that:

$$\frac{\mathbf{f(h)} - \mathbf{f(0)} - A(\mathbf{0})\mathbf{h}}{\|\mathbf{h}\|} = \frac{A(\mathbf{h})\mathbf{h} - \mathbf{0} - A(\mathbf{0})\mathbf{h}}{\|\mathbf{h}\|} \tag{3.97}$$

$$= \frac{(A(\mathbf{h}) - A(\mathbf{0}))\,\mathbf{h}}{\|\mathbf{h}\|} \tag{3.98}$$

$$\Rightarrow \frac{\|\mathbf{f(h)} - \mathbf{f(0)} - A(\mathbf{0})\mathbf{h}\|}{\|\mathbf{h}\|} \leq \frac{\|A(\mathbf{h}) - A(\mathbf{0})\|_i \|\mathbf{h}\|}{\|\mathbf{h}\|} \tag{3.99}$$

$$= \|A(\mathbf{h}) - A(\mathbf{0})\|_i \quad \forall \mathbf{h} \neq \mathbf{0} \tag{3.100}$$

so that:

$$\frac{\mathbf{f(h)} - \mathbf{f(0)} - A(\mathbf{0})\mathbf{h}}{\|\mathbf{h}\|} \to 0 \quad \text{as} \quad \mathbf{h} \to 0 \tag{3.101}$$

Q.E.D.

We are now in a position to prove our assertion that the continuous Ricatti design method will yield locally stable controllers, viz:

Lemma 3.3.2 *Given the system:*

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{3.102}$$

and feedback law

$$\mathbf{u} = -K(\mathbf{x})\mathbf{x} \tag{3.103}$$

where

$$K(\mathbf{x}) = R^{-1}B^T(\mathbf{x})P(\mathbf{x})\mathbf{x} \tag{3.104}$$

and $P(\mathbf{x})$ is the positive-semi-definite solution to:

$$0 = Q + P(\mathbf{x})A(\mathbf{x}) + A(\mathbf{x})^T P(\mathbf{x}) - P(\mathbf{x})B(\mathbf{x})R^{-1}B^T(\mathbf{x})P(\mathbf{x}) \tag{3.105}$$

Assume $A(\mathbf{x}), B(\mathbf{x})$ is stabilizable and $A(\mathbf{x}), Q^{\frac{1}{2}}$ is detectable. Then the closed loop

system will be locally exponentially stable.

Proof:

The closed loop dynamics of the system is given by:

$$\dot{\mathbf{x}} = \mathbf{f}_{cl}(\mathbf{x}) \tag{3.106}$$

$$= A_{cl}(\mathbf{x})\mathbf{x} \tag{3.107}$$

where:

$$A_{cl}(\mathbf{x}) = A(\mathbf{x}) - B(\mathbf{x})K(\mathbf{x}) \tag{3.108}$$

Since we have $A(\mathbf{0}), B(\mathbf{0})$ stabilizable and $A(\mathbf{0}), Q^{\frac{1}{2}}$ detectable we know from the properties of Linear Quadratic Regulators that $A_{cl}(\mathbf{0})$ will have all its poles strictly in the left half plane (see e.g. [32]). Furthermore from Lemma (3.3.1) we know that the Jacobian of $\mathbf{f}_{cl}(\mathbf{x})$ is given by $A_{cl}(0)$. Local exponential stability of the continuous Ricatti design method then follows from Theorem 3.3.1.

$$\text{\_\_\_\_\_} Q.E.D.$$

At this point we have rigorously established the local stability of the continuous Ricatti design method. In Section 3.3.3 we will further analyze the stability of the method and show why we expect the stability region for the continuous Ricatti design method to be large.

3.3.2 Single Equilibrium Point at the Origin

If we applied an arbitrary control law to the system (3.102) we might get multiple equilibrium points for the closed loop system which would in general be undesirable. However the following lemma shows that if we apply the continuous Ricatti design method we will have only a single equilibrium point at the origin.

Lemma 3.3.3 *Given the system, feedback law and stabilizability and detectability assumptions of Lemma 3.3.2, the closed loop system will have a single equilibrium point at* $\mathbf{x} = \mathbf{0}$

Proof:

The closed loop dynamics are given by equations (3.107),(3.108). Clearly at $\mathbf{x} = \mathbf{0}$ we will have $\dot{\mathbf{x}} = \mathbf{0}$. It remains to be shown that $\dot{\mathbf{x}} \neq \mathbf{0}$, $\forall \mathbf{x} \neq \mathbf{0}$. Since $\left\{A(\mathbf{x}), B(\mathbf{x}), Q^{\frac{1}{2}}\right\}$ is stabilizable and detectable and $K(\mathbf{x})$ is the feedback gain from the steady state regulator problem, the matrix $A_{cl}(\mathbf{x}) = (A(\mathbf{x}) - B(\mathbf{x})K(\mathbf{x}))$ will have all its eigenvalues in the open left half plane. Thus $A_{cl}(\mathbf{x})$ will have an empty nullspace and thus $A_{cl}(\mathbf{x})\mathbf{x} \neq \mathbf{0}$ $\forall \mathbf{x} \neq \mathbf{0}$.

Q.E.D.

3.3.3 Non-local Stability Analysis via Linear Time Varying Regulator Theory

The continuous Ricatti design method outlined in Section 3.1 can also be motivated from a stability standpoint by using results that are available for linear time varying systems. Our goal will be to provide a rationale of why we expect the continuous Ricatti design method to yield closed loop systems with large stability domains. The connection between (3.102) and linear time varying systems comes by noting that along a given trajectory $\mathbf{x}(t)$ the matrices $A(\mathbf{x}), B(\mathbf{x})$ become functions of time. So we can think of our quasilinear system as an ensemble of time varying systems "parameterized" by the trajectory $\mathbf{x}(t)$, viz:

$$\dot{\mathbf{x}} = A(\mathbf{x}(t))\mathbf{x} + B(\mathbf{x}(t))\mathbf{u} \tag{3.109}$$

$$\approx A(t)\mathbf{x} + B(t)\mathbf{u} \tag{3.110}$$

Chapter 4 will discuss a method which further exploits the analogy with linear time-varying systems. We should point out that because we are not dealing with a true linear time varying system the analysis we provide here does not guarantee stability, but only gives us some expectation of stable behavior.

In order to motivate the stability of the continuous Ricatti design method we need two results for linear time varying systems (LTV systems). The first result we need comes from Kokotovic et al [30]. Consider the following optimization problem for LTV systems:

For the system:

$$\dot{\mathbf{x}} = A(t)\mathbf{x} + B(t)\mathbf{u} \quad \text{with } \mathbf{x}(0) = \mathbf{x}_0 \tag{3.111}$$

find the control which minimizes:

$$J = \frac{1}{2}\int_0^T \left(\mathbf{x}^T C^T C\mathbf{x} + \mathbf{u}^T R\mathbf{u}\right) \mathrm{dt} \tag{3.112}$$

subject to:

$$\mathbf{x}(T) = \mathbf{x}_T \tag{3.113}$$

Reference [30] shows that if T is large with respect to the system dynamics and certain conditions outlined below are met, then the solution to the problem above exhibits "boundary layers". That is, the full optimal control can be approximated by the solution of two simpler problems, one at $t = 0$ and one at $t = T$. The following theorem slightly adapted from [30] makes these ideas more precise:

Theorem 3.3.2 *For the system (3.111) and cost-function (3.112), assume that:*

- $\dot{A}(t), \dot{B}(t)$ *are continuous for all* $t \in [0, T]$

- *The eigenvalues of the Hamiltonian matrix:*

$$\mathcal{H} = \begin{bmatrix} A(t) & -B(t)R^{-1}B(t) \\ -C^TC & -A^T(t) \end{bmatrix} \tag{3.114}$$

all lie off the imaginary axis.

- *The pair $A(t), B(t)$ satisfies the frozen controllability condition, that is:*

$$\text{rank}\left[B(t),\, A(t)B(t),\, \ldots,\, A^{n-1}(t)B(t)\right] = n \quad \forall t \in [0,T] \tag{3.115}$$

where $n = \dim(\mathbf{x})$ of the system.

Then there exists an $\epsilon_1 > 0$ such that for all $\epsilon \stackrel{\text{def}}{=} \frac{1}{T} \in (0, \epsilon_1]$ the optimal state trajectory and optimal control satisfies:

$$\mathbf{x}(t) = \mathbf{x}_l(t) + \mathbf{x}_r(\sigma) + O(\epsilon) \tag{3.116}$$

$$\mathbf{u}(t) = \mathbf{u}_l(t) + \mathbf{u}_r(t) + O(\epsilon) \tag{3.117}$$

where the initial and terminal boundary layer controls are given by:

$$\mathbf{u}_l(t) \stackrel{\text{def}}{=} -R^{-1}B^T(0)\lambda_l(t) \tag{3.118}$$

$$\mathbf{u}_r(t) \stackrel{\text{def}}{=} -R^{-1}B^T(T)\lambda_r(\sigma) \tag{3.119}$$

with:

$$\sigma \stackrel{\text{def}}{=} T - t \tag{3.120}$$

$$\lambda_l(t) \stackrel{\text{def}}{=} P(0)\mathbf{x}_l(t) \tag{3.121}$$

$$\lambda_r(\sigma) \stackrel{\text{def}}{=} N(T)\mathbf{x}_r(\sigma) \tag{3.122}$$

and

$$\frac{d\mathbf{x}_l}{dt} = \left[A(0) - B(0)R^{-1}B^T(0)P(0)\right]\mathbf{x}_l \quad \mathbf{x}_l|_{t=0} = \mathbf{x}_0 \tag{3.123}$$

$$\frac{d\mathbf{x}_r}{d\sigma} = -\left[A(T) - B(T)R^{-1}B^T(T)N(T)\right]\mathbf{x}_r \quad \mathbf{x}_r|_{\sigma=0} = \mathbf{x}_T \tag{3.124}$$

and $P(0)$ is the positive definite solution of the algebraic Ricatti equation:

$$0 = C^TC + \mathcal{P}(t)A(t) + A(t)^T\mathcal{P}(t) - \mathcal{P}(t)B(t)R^{-1}B^T(t)\mathcal{P}(t) \tag{3.125}$$

at $t = 0$, and $N(T)$ is the negative definite solution to the same equation at $t = T$.

The usefulness of this theorem is that it enables one to find an asymptotic approximation to the solution of the time-varying optimal control problem by solving two time invariant Linear Quadratic Regulator problems, one at $t = 0$ and one at $t = T$. The boundary layer notion can be explained as follows. Under the frozen controllability assumption (3.115) the dynamics of both equation (3.123) and equation (3.124) will be stable [40]. Hence for $T \gg 1$ we will have:

$$\mathbf{x}_l|_{t=T} \approx \mathbf{0} \tag{3.126}$$

$$\mathbf{x}_r|_{\sigma=T} \approx \mathbf{0} \tag{3.127}$$

Thus at $t \approx 0$:

$$\mathbf{x} \approx \mathbf{x}_l \tag{3.128}$$

$$\mathbf{u} \approx -R^{-1}B^T(0)P(0)\mathbf{x}_l \tag{3.129}$$

That is, the initial (boundary layer) behavior of the closed loop system is dominated by the dynamics of equation (3.123). Similarly at $t \approx T \Rightarrow \sigma \approx 0$

$$\mathbf{x} \approx \mathbf{x}_r \tag{3.130}$$

$$\mathbf{u} \approx -R^{-1}B^T(T)N(T)\mathbf{x}_r \tag{3.131}$$

So we see that the terminal (boundary layer) behavior of the closed loop system is dominated by the dynamics of equation (3.124). From equation (3.129) it also becomes clear that to approximately compute the initial control for the given problem we do not need to know the future values of $A(t), B(t)$ provided $T \rightarrow \infty$. (Note: If $\mathbf{x}_T \equiv \mathbf{0}$ then $\mathbf{x}_\tau \equiv \mathbf{0} \quad \forall t$ and the approximation will be even better). Furthermore the result implies that the control input (3.129) will be asymptotically correct for an ensemble of linear time varying systems that have the same value for $A(0), B(0)$.

The second result we need comes from receding horizon control theory (see Chapter 4 for more detailed discussion and references). Receding Horizon Control uses the following strategy:

1. At each time instant t solve a linear quadratic regulator problem over a horizon starting at the current time t and ending at the time $t + T$ subject to the terminal constraint $\mathbf{x}(t + T) = \mathbf{0}$.

2. Then at each time instant t apply the initial control, i.e. $\mathbf{u}(t)$, found from the regulator problem that we have just solved.

The following theorem slightly adapted from [36] summarizes the fundamental stability result for receding horizon control.

Theorem 3.3.3 *Let* $\mathbf{u}^*(\tau), \tau \in [t, t + T]$ *be the control time history which minimizes the cost:*

$$J = \frac{1}{2} \int_t^{t+T} \left(\mathbf{x}^T C^T C \mathbf{x} + \mathbf{u}^T R \mathbf{u} \right) \mathrm{d}\tau \tag{3.132}$$

for the system (3.111) subject to the terminal constraint $\mathbf{x}(t + T) = \mathbf{0}$. *Assume that:*

- R *is positive definite*
- $Q \stackrel{\text{def}}{=} C^T C$ *is positive semi-definite*
- *the pair* $A(t), B(t)$ *is uniformly completely controllable for some* $T_c > 0$

Then

1. *If we choose $T \geq T_c$ and at each time t we apply the control $\mathbf{u} = \mathbf{u}^*(t)$ then the closed loop system will be uniformly asymptotically stable.*

2. *The control $\mathbf{u}^*(t)$ is given by:*

$$\mathbf{u}^*(t) = -R^{-1}B^T(t)P(t,t+T)\mathbf{x}(t) \tag{3.133}$$

where $P(t,t+T) = M^{-1}(t,t+T)$ and $M(t,t+T)$ is found by integrating the time-varying Ricatti Equation:

$$\begin{aligned} -\frac{\partial M(\tau,t+T)}{\partial \tau} &= B(\tau)R^{-1}B^(\tau) - M(\tau,t+T)A^T(\tau) - A(\tau)M(\tau,t+T) \\ &\quad -M(\tau,t+T)C^TCM(\tau,t+T) \end{aligned} \tag{3.134}$$

backwards from $\tau = t+T$ to $\tau = t$ subject to:

$$M(t+T,t+T) = 0 \tag{3.135}$$

Note that the receding horizon control method is no longer optimal for the original cost function (3.132) (see [35] for a bound on the cost incurred by the receding horizon control law).

By combining Theorems (3.3.2) and (3.3.3) we can now provide a stability argument for the case where we are applying the continuous Ricatti design method to *linear time-varying systems*, viz:

- From Theorem (3.3.3) we know that the system (3.111) will be stable if we apply the receding horizon control described above.
- In stead of finding the control (3.133) by integrating the matrix Ricatti differential equation (3.134) over the interval $[t, t+T]$ we can find an asymptotically

correct value (as $T \rightarrow \infty$) for $\mathbf{u}^*(t)$ by using Theorem (3.3.2). The approximate value of $\mathbf{u}^*(t)$ is given by:

$$\mathbf{u}^*(t) = -R^{-1}B^T(t)P(t)\mathbf{x}(t) \tag{3.136}$$

where $P(t)$ is the positive definite solution to:

$$0 = C^TC + P(t)A(t) + A^T(t)P(t) - P(t)B(t)R^{-1}B^T(t)P(t) \tag{3.137}$$

So we expect stable behavior for linear time-varying systems insofar as the asymptotic approximation is valid.

We see that the procedure outlined above amounts to the continuous Ricatti design method as applied to Linear Time Varying Systems. To motivate our expectation of closed loop stability for the case where we apply the continuous Ricatti design method to nonlinear systems, we treat the quasilinear system (3.109) as if it were a linear time varying system along a given solution trajectory as in equation (3.110). Here we exploit the fact that to find the asymptotically correct value of the receding horizon control input ($\mathbf{u}^*$ in equation (3.133)) we only need to know the current values of $A(\mathbf{x}(t)), B(\mathbf{x}(t))$. However note that while we can generally assume that $A(t), B(t)$ are well-behaved for the true linear time-varying case, we have no such guarantees for $A(\mathbf{x}(t)), B(\mathbf{x}(t))$ and hence cannot guarantee stability for the case when we are dealing with nonlinear systems.

3.4 Stability Region Assessment

In the previous sections we have analyzed the stability properties of the continuous Ricatti design method and have shown rigorously that the closed loop systems will be locally stable. We also indicated why we expect the stability domains to be relatively large. In this section we will examine computational methods to assess the size of

the stability domain. The first method will provide a straightforward procedure to assess a lower bound on the size of the stability domain while the second method will provide a computationally more complex procedure to determine the exact extent of the stability domain. Although not discussed here further we should point out that simulating the closed loop dynamics still provides a very useful method for assessing the stability of nonlinear systems.

3.4.1 Stability Domain Assessment Using Quadratic Forms

The first method we use to assess the stability domain of the nonlinear closed loop system will take a Lyapunov function approach and will provide a lower bound on the size of the stability domain (see also [21], [64] for related results). In order to develop the method we will use La Salle's theorem [37] which is stated below. Before stating La Salle's theorem we will need the following definitions (see [59]):

Definition 3.4.1 *1. A scalar continuous function $V(\mathbf{x})$ is said to be* **locally positive definite** *if $V(\mathbf{0}) = 0$ and in an open neighborhood of the origin:*

$$\mathbf{x} \neq \mathbf{0} \Rightarrow V(\mathbf{x}) > 0 \tag{3.138}$$

2. If $V(\mathbf{0}) = \mathbf{0}$ and the above property holds over the whole state space, then $V(\mathbf{x})$ is said to be **globally positive definite**.

3. A scalar continuous function $V(\mathbf{x})$ is said to be **negative definite** *if $-V(\mathbf{x})$ is positive definite.*

Theorem 3.4.1 *(***La Salle***) Consider an autonomous system of the form:*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{3.139}$$

with $\mathbf{f}(\mathbf{x})$ *continuous and* $\mathbf{f}(\mathbf{0}) = \mathbf{0}$. *Let* $V(\mathbf{x})$ *be a locally positive definite function with continuous first partial derivatives and suppose that:*

- *for some* $l > 0$ *the set:*

$$\Omega_l \stackrel{\text{def}}{=} \{\mathbf{x} | V(\mathbf{x}) \leq l\} \tag{3.140}$$

  is bounded.

- V *is bounded below on* Ω_l

- $\dot{V}(\mathbf{x}) \leq 0 \quad \forall \mathbf{x} \in \Omega_l$

- *the set:*

$$\mathcal{S} \stackrel{\text{def}}{=} \{\mathbf{x} \in \Omega_l | \dot{V}(\mathbf{x}) = 0\} \tag{3.141}$$

  contains no trajectories of (3.139) other than the trivial trajectory $\mathbf{x}(t) \equiv 0$.

Then the equilibrium point $\mathbf{0}$ *of (3.139) is asymptotically stable.*

The above theorem provides us with sufficient conditions that guarantee that trajectories starting in a domain Ω_l will tend to $\mathbf{0}$ as $t \rightarrow \infty$ provided that we can find a suitable function $V(\mathbf{x})$. The following corollary shows how we can always construct a function $V(\mathbf{x})$ that satisfies the conditions of La Salle's theorem and provides us with a well defined region Ω_l if the linearized version of (3.139) is locally asymptotically stable (note that Lyapunov's linearization method, Theorem 3.3.1, only states that some stable neighborhood exists but does not show how to determine it).

Corollary 3.4.1 *Consider a system:*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{3.142}$$

with $\mathbf{f}(\mathbf{x})$ *continuously differentiable and* $\mathbf{f}(\mathbf{0}) = \mathbf{0}$. *Let*

$$A \stackrel{\text{def}}{=} \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{0}} \tag{3.143}$$

Assume that A has all its eigenvalues in the open left half plane and let P_0 be the positive definite solution to the matrix Lyapunov equation:

$$P_0 A + A^T P_0 = -I \tag{3.144}$$

(Note: $P_0 > 0$ is guaranteed to exist, since A is strictly stable see [59]). Define:

$$V(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{2}\mathbf{x}^T P_0 \mathbf{x} \tag{3.145}$$

$$\mathcal{S} \stackrel{\text{def}}{=} \{\mathbf{x} | \dot{V} = \frac{\partial V}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}) \equiv 0\} \tag{3.146}$$

and let l be the lower bound of the values of $V(\mathbf{x})$ on the set $\mathcal{S}$ with the exception of the origin (if $\mathcal{S}$ contains only the origin set $l = \infty$). Then the origin will be asymptotically stable for every solution originating in the region Ω_{l_ε} defined by:

$$\Omega_{l_\varepsilon} \stackrel{\text{def}}{=} \{\mathbf{x} | V(\mathbf{x}) \leq l - \varepsilon\} \tag{3.147}$$

where $\varepsilon > 0$.

Proof:

We will prove this corollary by showing that when l is finite we satisfy the requirements of Theorem (3.4.1) and that when l is infinite, (3.142) will be globally asymptotically stable by application of Lyapunov's direct method [59].

Before proceeding we need the following from [63]:

- Claim: $\dot{V}(\mathbf{x})$ is a (locally) negative definite function

  Proof of Claim:

  We can write:

$$\mathbf{f}(\mathbf{x}) = A\mathbf{x} + \mathbf{r}(\mathbf{x}) \tag{3.148}$$

with $\mathbf{r}(\mathbf{0}) = \mathbf{0}$ since $A\mathbf{x}$ is the first term in the Taylor expansion of $\mathbf{f}(\mathbf{x})$. Now:

$$\begin{aligned}
\dot{V} &= \mathbf{x}^T P_0 \mathbf{f}(\mathbf{x}) && (3.149)\\
&= \mathbf{x}^T P_0 \left(A\mathbf{x} + \mathbf{r}(\mathbf{x})\right) && (3.150)\\
&= \frac{1}{2}\mathbf{x}^T \left(P_0 A + A^T P_0\right)\mathbf{x} + \mathbf{x}^T P_0 \mathbf{r}(\mathbf{x}) && (3.151)\\
&= -\frac{1}{2}\mathbf{x}^T\mathbf{x} + \mathbf{x}^T P_0 \mathbf{r}(\mathbf{x}) && (3.152)
\end{aligned}$$

Now

$$\mathbf{x}^T P_0 \mathbf{r}(\mathbf{x}) \leq \|\mathbf{x}\|\bar{\sigma}\|\mathbf{r}(\mathbf{x})\| \qquad (3.153)$$

where $\bar{\sigma}$ is the maximum eigenvalue of P_0. Since $\mathbf{r}(\mathbf{x})$ is the remainder term of the Taylor expansion of $\mathbf{f}(\mathbf{x})$ we know that:

$$\lim_{\|\mathbf{x}\|\to 0} \frac{\|\mathbf{r}(\mathbf{x})\|}{\|\mathbf{x}\|} = 0 \qquad (3.154)$$

and hence we can find an open neighborhood of the origin say $\mathcal{N}$ such that:

$$\|\mathbf{r}(\mathbf{x})\| \leq \frac{1}{4\bar{\sigma}}\|\mathbf{x}\| \quad \forall \mathbf{x} \in \mathcal{N} \qquad (3.155)$$

Hence

$$\mathbf{x}^T P_0 \mathbf{r}(\mathbf{x}) \leq \frac{1}{4}\|\mathbf{x}\|\|\mathbf{x}\| \quad \forall \mathbf{x} \in \mathcal{N} \qquad (3.156)$$

and subsequently

$$\dot{V} \leq -\frac{1}{2}\mathbf{x}^T\mathbf{x} + \frac{1}{4}\|\mathbf{x}\|^2 \quad \forall \mathbf{x} \in \mathcal{N} \qquad (3.157)$$

so that $\dot{V}$ is locally negative definite around the origin.

Let us now consider the case where l is finite. In this case $\Omega_{l\varepsilon}$ will be a finite hyper-ellipsoid centered at the origin since P_0 is positive definite and thus:

$$V(\mathbf{x}) = \mathbf{x}^T P_0 \mathbf{x} < l \quad \forall \mathbf{x} \in \Omega_{l\varepsilon} \qquad (3.158)$$

Furthermore:

$$\dot{V}(\mathbf{x}) \neq 0 \quad \forall \mathbf{x} \in \Omega_{l\varepsilon}, \quad \mathbf{x} \neq \mathbf{0} \tag{3.159}$$

(otherwise we could find $\mathbf{x}^*$ such that $V(\mathbf{x}^*) < l$ and $\dot{V}(\mathbf{x}^*) = 0$ which would contradict our assumption that l is a lower bound). Since $\dot{V}$ is negative definite in a neighborhood of the origin and $\dot{V}$ is continuous we see from the intermediate value theorem [46] that $\dot{V} \leq 0 \quad \forall \mathbf{x} \in \Omega_l$. Asymptotic stability then follows from Theorem (3.4.1) since $\mathcal{S} = \{\mathbf{x} \in \Omega_\varepsilon | \dot{V}(\mathbf{x}) = 0\}$ contains only the origin.

The remaining case we have to consider is when l is infinite, i.e. $\dot{V} \neq 0 \quad \forall \mathbf{x} \neq \mathbf{0}$. Since $\dot{V}$ is at least locally negative definite and $\dot{V}$ is continuous it then follows from the intermediate value theorem that $\dot{V} < 0 \quad \forall \mathbf{x} \neq \mathbf{0}$. Global asymptotic stability then follows from Lyapunov's direct method by noting that $V(\mathbf{x}) = \mathbf{x}^T P_0 \mathbf{x}$ is radially unbounded [59].

Q.E.D.

With Corollary (3.142) in place we can now specify a computational procedure which will give a lower bound on the size of the stability region for a closed loop nonlinear system, i.e. a system of the form (3.142). To compute an estimate of the stability region we do the following

1. Compute:

$$A = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{0}} \tag{3.160}$$

2. Solve the matrix Lyapunov equation:

$$P_0 A + A^T P_0 = -I \tag{3.161}$$

3. Find the minimum value of $V(\mathbf{x})$ such that $V(\mathbf{x}) \neq \mathbf{0}$ and $\dot{V}(\mathbf{x}) = 0$. Some possibilities for doing this are:

(a) Compute V and $\dot{V}$ for several points along "rays" emanating from the origin to find $l \stackrel{\text{def}}{=}$ "smallest value of V where $\dot{V} = 0$".

(b) Use a single variable rootfinding algorithm along "rays" emanating from the origin to find the first location where $\dot{V} = 0$ along the ray. Among these locations find the value of $\mathbf{x}$ that minimizes V

(c) Use a multivariable rootfinding algorithm to find locations where $\dot{V} = 0$. Among these locations find the value of $\mathbf{x}$ that minimizes V.

4. Once we have found the desired minimum value of V, say V_{min}, we know from Corollary (3.142) that all the trajectories starting within the hyper-ellipsoid defined by $\mathbf{x}^T P_0 \mathbf{x} = V_{min}$ will tend to $\mathbf{0}$ as $t \to \infty$

Example 3.4.1 As an example of using this approach consider the following example system from Hahn [21]:

$$\dot{x} = -x + 2x^2 y \tag{3.162}$$

$$\dot{y} = -y \tag{3.163}$$

For this example it is possible to exactly determine the size of the stability domain (see [21] and Section 3.4.2). It can be shown that all trajectories starting in the region defined by

$$y < \frac{1}{x}, \quad x > 0 \tag{3.164}$$

$$y > \frac{1}{x}, \quad x < 0 \tag{3.165}$$

$$y \in (-\infty, \infty), \quad \text{for} \quad x = 0 \tag{3.166}$$

will tend to the origin as $t \to \infty$.

Let us now use the quadratic form method outlined above. We get:

$$A = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{0}} \tag{3.167}$$

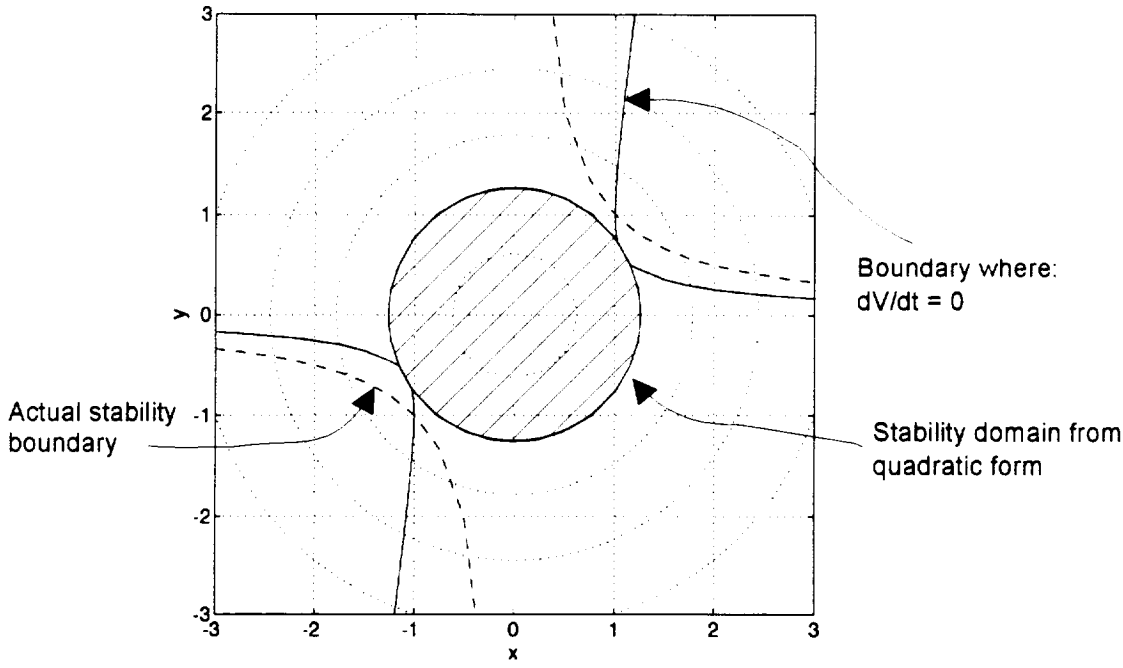


Figure 3-4: Stability Domain Using Quadratic Form

$$= \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \tag{3.168}$$

while P_0 from equation (3.144) evaluates to:

$$P_0 = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \tag{3.169}$$

Note that for this example the contours of constant V are circles. Searching for locations where $\dot{V} = 0$ we get the results shown in Figure 3-4. The stability domain we find using this method is significantly smaller than the actual stability domain because it is constrained to be an ellipsoid.

3.4.2 Stability Domain Assessment using Zubov's Method

In this section we will discuss Zubov's method (see e.g. [21], [64]) which can be used to determine the exact domain of stability for nonlinear systems of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{3.170}$$

Zubov's method has two important advantages for stability analysis:

1. It provides a deterministic method to find Lyapunov functions for non-linear systems, as opposed to the usual procedure of "guessing" candidate Lyapunov functions.

2. It provides a method to exactly determine the extent of the stability domain. This is different to standard Lyapunov theory which tends to provide results that either:

   - guarantee the existence of some stable ε-neighborhood of an equilibrium point but give no indication of the extent of the stability domain.
   - guarantee that systems are globally stable. (finding Lyapunov functions for this case is notoriously difficult).

The advantages come with a price however. It requires one to solve a partial differential equation (P.D.E.). Zubov's main theorem makes this more precise [64], [21]:

Theorem 3.4.2 *Let $\mathcal{U}$ be an open domain of the state space and let $\bar{\mathcal{U}}$ be its closure; $\mathcal{U}$ shall contain the origin. Necessary and sufficient conditions for $\mathcal{U}$ to be the exact domain of attraction of the equilibrium of*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \quad (\mathbf{f}(\mathbf{0}) = \mathbf{0}) \tag{3.171}$$

is the existence of two functions $v(\mathbf{x})$ and $\varphi(\mathbf{x})$ with the following properties:

1. *$v(\mathbf{x})$ is defined and continuous in $\mathcal{U}$, $\varphi(\mathbf{x})$ is defined and continuous in the whole space*

2. *$\varphi(\mathbf{x})$ is positive definite for all $\mathbf{x}$*

3. *$v(\mathbf{x})$ is negative definite, and for $\mathbf{x} \in \mathcal{U}$, $\mathbf{x} \neq \mathbf{0}$, the inequality $0 < v(\mathbf{x}) < 1$ holds.*

4. *If $\mathbf{y} \in \bar{\mathcal{U}} - \mathcal{U}$, then $\lim_{\mathbf{x}\to\mathbf{y}} v(\mathbf{x}) = 1$, furthermore $\lim_{|\mathbf{x}|v(\mathbf{x})\to\infty} = 1$ provided that this limit process can be carried out for $\mathbf{x} \in \mathcal{U}$.*

5.
$$\frac{\partial v}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}) = \varphi(\mathbf{x})(v(\mathbf{x}) - 1) \tag{3.172}$$

The following slightly modified example from Hahn [21] shows how the theorem can be applied by analytically finding a solution to (3.172).

Example 3.4.2 Consider the system:

$$\dot{x} = -x + 2x^2y \tag{3.173}$$

$$\dot{y} = -y \tag{3.174}$$

For this example, Zubov's P.D.E. (i.e. equation 3.172) becomes:

$$\frac{\partial v}{\partial x}(-x + 2x^2y) + \frac{\partial v}{\partial y}(-y) = \varphi(x, y)(v - 1) \tag{3.175}$$

If we set $\varphi(x, y)$ to the positive definite function:

$$\varphi(x, y) = \left(\frac{x}{\alpha}\right)^2 + \left(\frac{y}{\alpha}\right)^2 \tag{3.176}$$

we can find the following solution to equation (3.175):

$$v(x, y) = 1 - \exp\left(-\frac{1}{2}\left(\frac{y}{\alpha}\right)^2 - \frac{1}{2}\left(\frac{x}{\alpha}\right)^2\left(\frac{1}{1 - xy}\right)\right) \tag{3.177}$$

We see that the curve $xy = 1$ defines the stability boundary.

Numerical Solution of Zubov's Equation using Finite Differences

The example above demonstrates Zubov's method, but it also shows that in general we will have difficulty in obtaining exact analytical solutions. We therefore consider a numerical solution procedure to the example above. Several methods for solving P.D.E.'s exist, such as finite difference, finite elements etc. (see e.g. [1]). Perhaps the simplest approach to solving (3.172) is to use a finite difference method which we shall use below.

The usual approach to solving the P.D.E. (3.172) by finite difference methods will be to represent the unknown function, $v(\mathbf{x})$ in our case, by its values at a discrete grid of reference points. For example, for a second order system we could use:

$$x_i = x_0 + ih \tag{3.178}$$

$$y_j = y_0 + jh \tag{3.179}$$

as shown in Figure (3-5). Such an approach however rapidly becomes unmanageable if we attempt to solve the P.D.E. over the whole domain of interest at once. For example, say we want to solve Zubov's P.D.E. for an 8'th order system over a grid of 10 points per dimension. This means that we need to solve for the values of v at 10^8 points which translates into the numerical challenge of solving a set of 10^8 linear equations. Therefore we propose to solve the P.D.E. for smaller domains at a time. A straightforward way to accomplish this is to solve the P.D.E. along narrow solution strips as shown in Figure (3-6). To assess the size of the stability domain we "point" these solution strips into all the directions of interest. Note that this approach has the added advantage that it is amenable to parallelization.

Let us now develop a finite difference solution for Zubov's equation in more detail. We discuss the method as applied to a second order system noting that the same

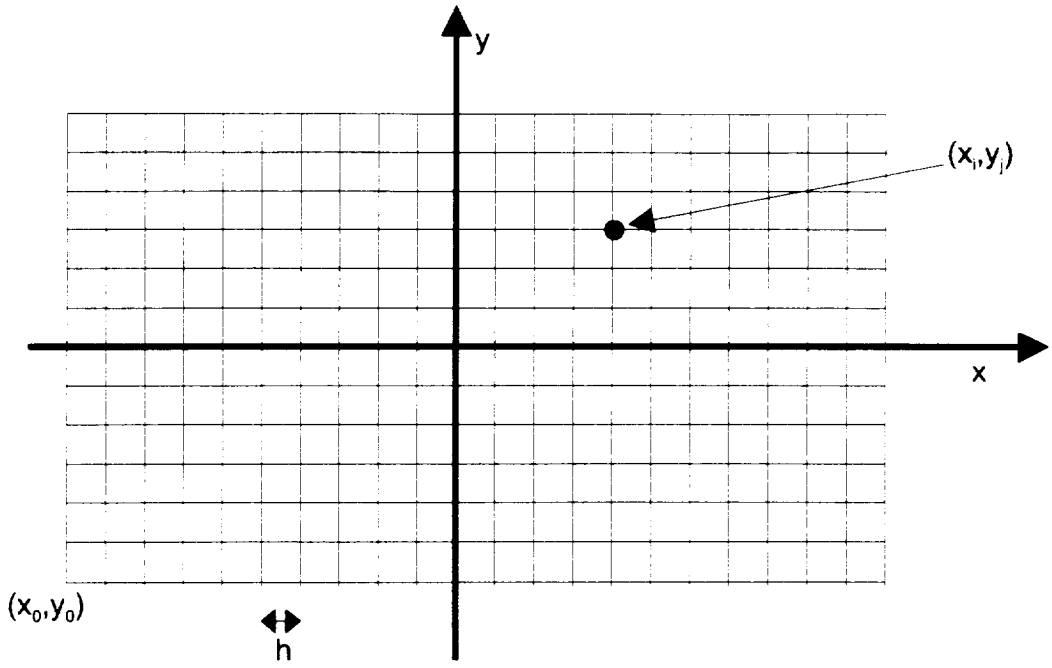


Figure 3-5: Standard Grid for Finite Difference Method

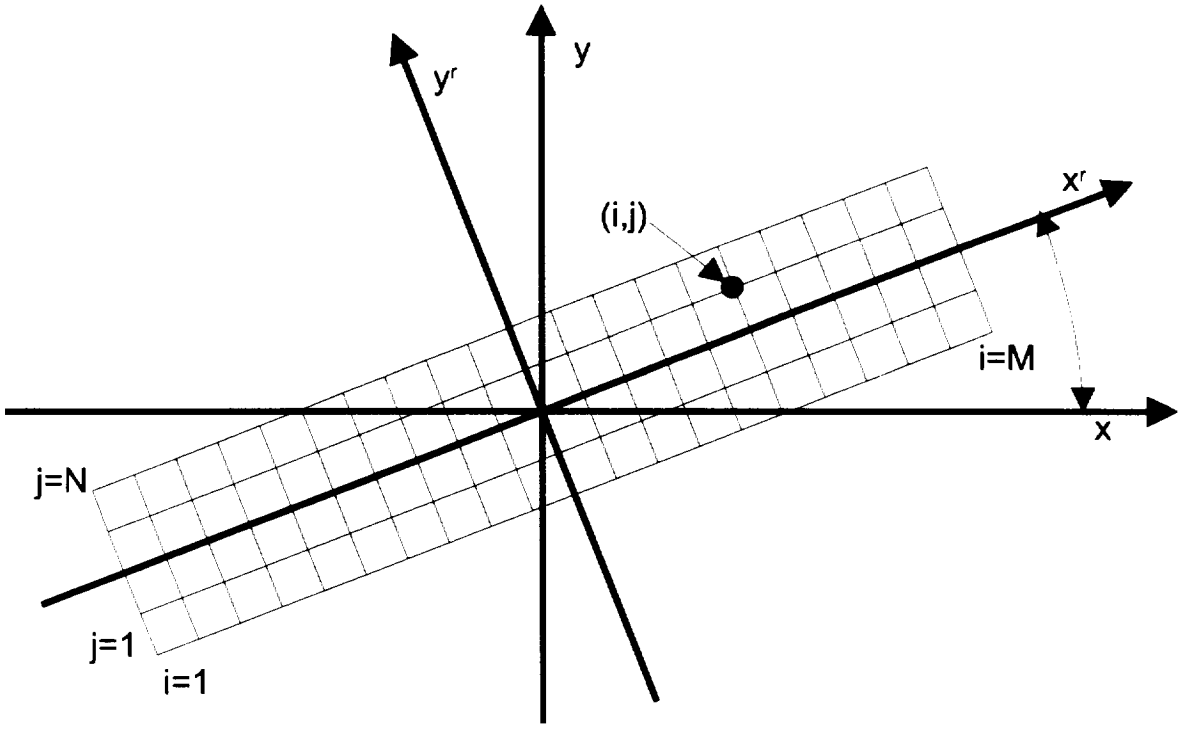


Figure 3-6: Solution Strips for Zubov's P.D.E.

procedure can be used for higher order systems.

Figure (3-6) shows a general solution strip that is rotated by an angle θ from the nominal x, y-axes. Let:

- $\mathbf{x}^r$ denote the co-ordinates of a point in rotated co-ordinates
- $\mathbf{x}$ denote the co-ordinates of the same point in non-rotated co-ordinates

Zubov's equation in rotated co-ordinates then becomes:

$$\frac{\partial v}{\partial \mathbf{x}^r}\dot{\mathbf{x}}^r = \varphi(v-1) \tag{3.180}$$

To evaluate $\dot{\mathbf{x}}^r$ we note that the transformation from non-rotated to rotated co-ordinates is a simple linear transformation, viz:

$$\mathbf{x}^r = C\mathbf{x} \tag{3.181}$$

where C is the direction cosine matrix relating the two sets of axes. For our 2-D example we will have:

$$\mathbf{x}^r = \begin{bmatrix} x^r \\ y^r \end{bmatrix}, \quad C = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{3.182}$$

So that the system dynamics in rotated co-ordinates becomes:

$$\begin{aligned} \dot{\mathbf{x}}^r &= C\dot{\mathbf{x}} && (3.183) \\ &= C\mathbf{f}(C^{-1}\mathbf{x}^r) && (3.184) \\ &\stackrel{\text{def}}{=} \mathbf{f}^r(\mathbf{x}^r) && (3.185) \end{aligned}$$

At this point we are in a position to set up the finite-difference approximations for Zubov's equation. We develop the approximations in rotated co-ordinates because of the simplification we obtain when the solution strip becomes a rectangular grid

aligned with the x^r axis (see Figure (3-6)). Let us denote the point:

$$\begin{bmatrix} x^r \\ y^r \end{bmatrix} = \begin{bmatrix} x_0^r + ih \\ y_0^r + jh \end{bmatrix} \tag{3.186}$$

on the grid, by the pair (i, j). Similarly let:

$$v(x^r, y^r) \equiv V_{i,j}, \quad \mathbf{f}^r(\mathbf{x}^r) \equiv \begin{bmatrix} f_1^r(x^r, y^r) \\ f_2^r(x^r, y^r) \end{bmatrix} \equiv \begin{bmatrix} f_{1_{i,j}}^r \\ f_{2_{i,j}}^r \end{bmatrix}, \quad \varphi(x^r, y^r) \equiv \varphi_{i,j} \tag{3.187}$$

Using this notation we get the following finite difference approximations:

- For a point, say (i, j), interior to the grid:

$$\left.\frac{\partial v}{\partial x^r}\right|_{i,j} \approx \frac{V_{i+1,j} - V_{i-1,j}}{2h} \tag{3.188}$$

$$\left.\frac{\partial v}{\partial y^r}\right|_{i,j} \approx \frac{V_{i,j+1} - V_{i,j-1}}{2h} \tag{3.189}$$

- For points on the edges we use one-sided finite-difference approximations as appropriate. For example for a point, say (i, j), on the left edge away from the corners:

$$\left.\frac{\partial v}{\partial x^r}\right|_{i,j} \approx \frac{V_{i+1,j} - V_{i,j}}{h} \tag{3.190}$$

$$\left.\frac{\partial v}{\partial y^r}\right|_{i,j} \approx \frac{V_{i,j+1} - V_{i,j-1}}{2h} \tag{3.191}$$

Using these approximations, Zubov's equation becomes a set of linear equations in the variables $V_{i,j}, i = 1 \ldots M, j = 1 \ldots N$, one equation for each grid point. For example at a point (i, j) interior to the grid, we get:

$$\left(\frac{V_{i+1,j} - V_{i-1,j}}{2h}\right) f_{1_{i,j}}^r + \left(\frac{V_{i,j+1} - V_{i,j-1}}{2h}\right) f_{2_{i,j}}^r = \varphi_{i,j}(V_{i,j} - 1) \tag{3.192}$$

which becomes a linear equation in the unknowns $V_{i-1,j}, V_{i,j-1}, V_{i,j}, V_{i,j+1}, V_{i+1,j}$, viz:

$$-\frac{f^r_{1_{i,j}}}{2h}V_{i-1,j} - \frac{f^r_{2_{i,j}}}{2h}V_{i,j-1} - \varphi_{i,j}V_{i,j} + \frac{f^r_{2_{i,j}}}{2h}V_{i,j+1} + \frac{f^r_{1_{i,j}}}{2h}V_{i+1,j} = -\varphi_{i,j} \tag{3.193}$$

Similar results hold for other points at the edges of the grid. Assembling all these equations we get a set of linear equations which we have to solve:

$$A\mathbf{v} = -\mathbf{b} \tag{3.194}$$

In equation (3.194):

$$\mathbf{v}^T \stackrel{\text{def}}{=} \left[V_{1,1} \;\; V_{1,2} \;\; \ldots \;\; V_{1,N} \;\; V_{2,1} \;\; \ldots \;\; V_{2,N} \;\; \ldots \;\; V_{M,1} \;\; \ldots \;\; V_{M,N} \right] \tag{3.195}$$

$$\mathbf{b}^T \stackrel{\text{def}}{=} \left[\varphi_{1,1} \;\; \varphi_{1,2} \;\; \ldots \;\; \varphi_{1,N} \;\; \varphi_{2,1} \;\; \ldots \;\; \varphi_{2,N} \;\; \ldots \;\; \varphi_{M,1} \;\; \ldots \;\; \varphi_{M,N} \right] \tag{3.196}$$

and each row in A corresponds to an equation such as (3.193). In general the matrix A will be sparse as can be seen by examining the row corresponding to equation (3.193):

$$\left[0 \;\; \ldots \;\; 0 \;\; -\frac{f^r_{1_{i,j}}}{2h} \;\; 0 \;\; \ldots \;\; 0 \;\; -\frac{f^r_{2_{i,j}}}{2h} \;\; -\varphi_{i,j} \;\; \frac{f^r_{2_{i,j}}}{2h} \;\; 0 \;\; \ldots 0 \;\; \frac{f^r_{1_{i,j}}}{2h} \;\; 0 \;\; \ldots \;\; 0 \right] \tag{3.197}$$

Exploiting the fact that A is sparse will enable us to further reduce our memory requirements [9].

Example 3.4.3 Consider again example 3.4.2 for which we know the exact solution. Using the finite difference method outlined above to solve Zubov's P.D.E. for this case we get the results shown in figures (3-7) and (3-8). The plots compare the values of the finite difference solution and the exact solution along a solution strip — values are plotted in rotated co-ordinates, i.e. along the x^r-axis of Figure 3-6. We used a solution strip as in Figure 3-6 with the following parameters:

$$\theta = 30^\circ \tag{3.198}$$

$$h = 0.05 \tag{3.199}$$

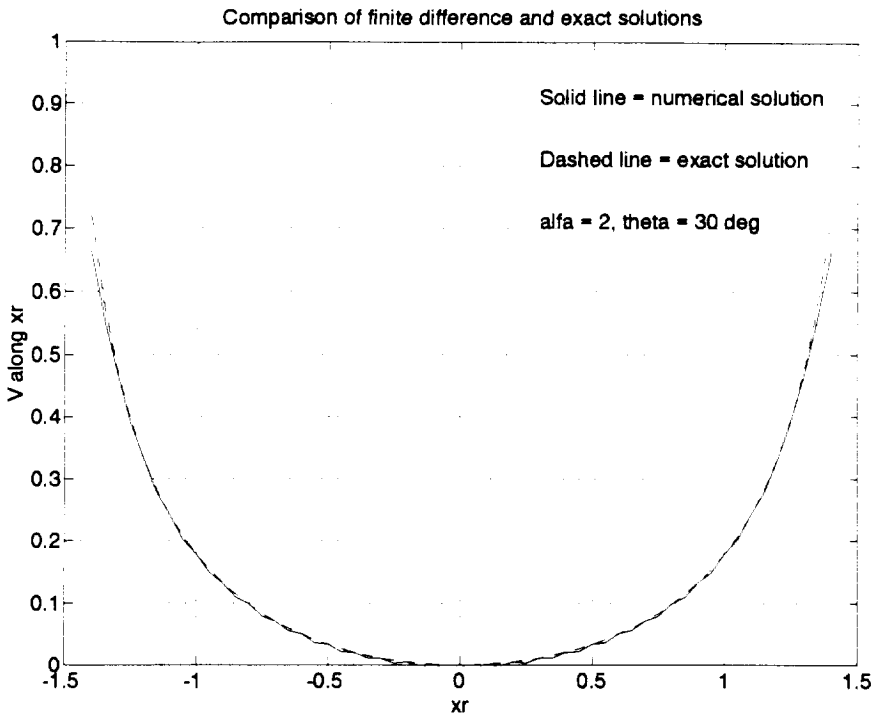


Figure 3-7: Exact vs. Finite Difference Solution to Zubov's P.D.E.

$$M = 56 \tag{3.200}$$

$$N = 10 \tag{3.201}$$

We used φ as in example 3.4.2 with:

$$\alpha = 2 \tag{3.202}$$

Numerical Conditioning of the Solution to Zubov's Equation

The approach outlined above to solve Zubov's P.D.E. has yielded promising results but further work is required to analyze the numerical conditioning of solutions. For instance in example 3.4.2 the exact solution to Zubov's P.D.E. is not well behaved. We see from equation (3.177) that when $xy \leq 1$ we have $0 \leq v(x,y) \leq 1$. However for $xy = 1 + \varepsilon$ with $0 < \varepsilon \ll 1$ we see that $v(x,y)$ will be large and negative. This

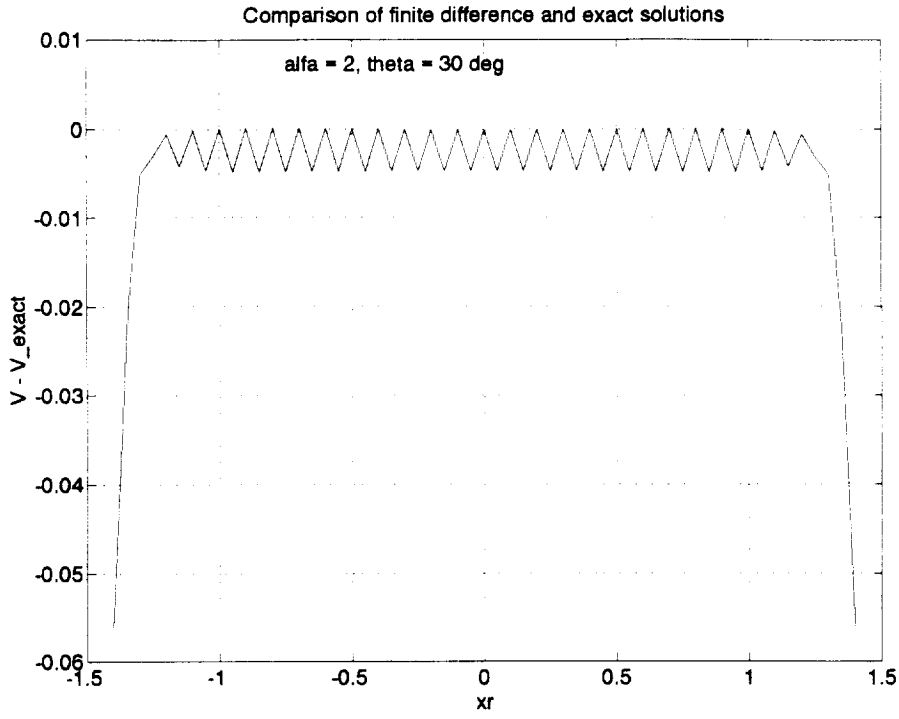


Figure 3-8: Error between Exact and Finite Difference Solutions to Zubov's P.D.E.

discontinuity in the (true) solution to the P.D.E. will generally cause difficulty for numerical solution methods.

A further issue is that the choice of $\varphi(x, y)$ also affects the behavior of the solution as explained in the following.

Zubov's equation can be written as:

$$\frac{\partial v}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}) = \varphi(\mathbf{x})(v(\mathbf{x}) - 1) \tag{3.203}$$

Along solution trajectories of $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ this equation can be written as:

$$\frac{\mathrm{d}v}{\mathrm{d}t} = \varphi(\mathbf{x})(v - 1) \tag{3.204}$$

or in reverse time τ:

$$\frac{\mathrm{d}v}{\mathrm{d}\tau} = -\varphi(\mathbf{x})(v - 1) \tag{3.205}$$

So along a specific trajectory $\mathbf{x}(\tau)$ in reverse time we get:

$$\frac{\mathrm{d}v}{\mathrm{d}\tau} = -\varphi(\mathbf{x}(\tau))(v-1) \tag{3.206}$$

which has a solution of the form:

$$v = 1 + c\exp^{-\int \varphi \mathrm{d}\tau} \quad c = \text{constant} \tag{3.207}$$

The influence of φ (which is a positive definite function) now becomes clearer. For example if φ is relatively small then v will tend to remain small and grow quickly at the stability boundary where it must attain the value $v = 1$. On the other hand if φ is relatively large then v will quickly approach its asymptotic value and it will numerically not be clear whether we are close to the stability boundary.

Example 3.4.4 Figures 3-9 and 3-10 show the results of redoing Example (3.4.3) where the only change has been in the parameter α of the function $\varphi(x, y)$ of equation (3.176). Using $\alpha = 0.1$ (i.e. φ relatively large) we get the results in Figure 3-9 which confirm that v quickly approaches its asymptotic value. Conversely when $\alpha = 10$ (i.e. φ relatively small), we see in Figure (3-10) that v remains relatively small. Note that in both cases the finite difference method produced accurate results.

3.5 Computational Issues

The continuous Ricatti design method requires in principle that we compute a feedback gain:

$$K(\mathbf{x}) = R^{-1}B^T(\mathbf{x})P(\mathbf{x}) \tag{3.208}$$

by first solving a steady state Ricatti equation:

$$0 = Q + P(\mathbf{x})A(\mathbf{x}) + A(\mathbf{x})^T P(\mathbf{x}) - P(\mathbf{x})B(\mathbf{x})R^{-1}B^T(\mathbf{x})P(\mathbf{x}) \tag{3.209}$$

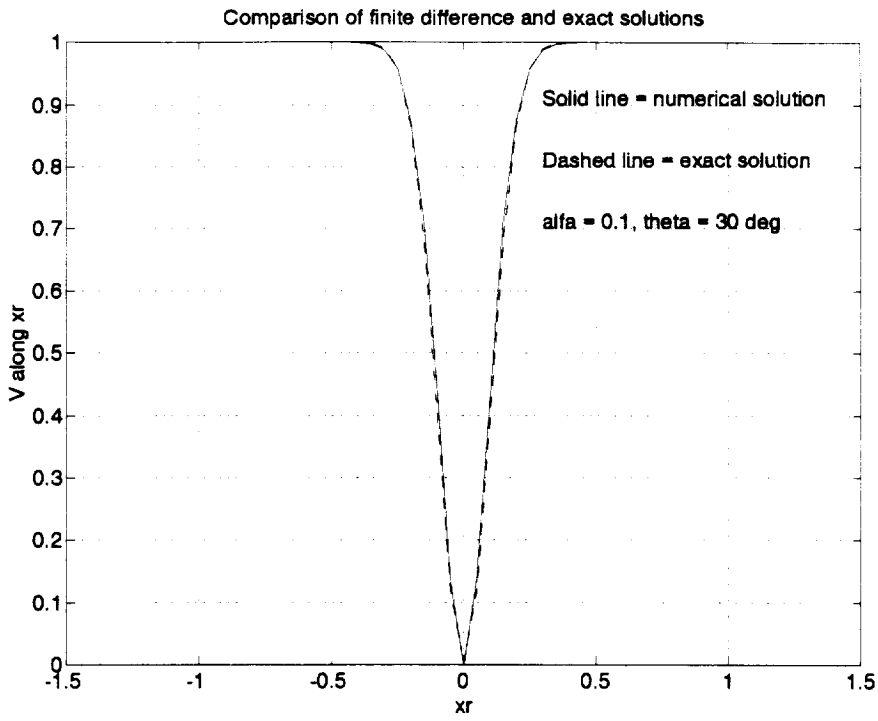


Figure 3-9: Exact vs. Finite Difference Solution to Zubov's P.D.E. — φ large

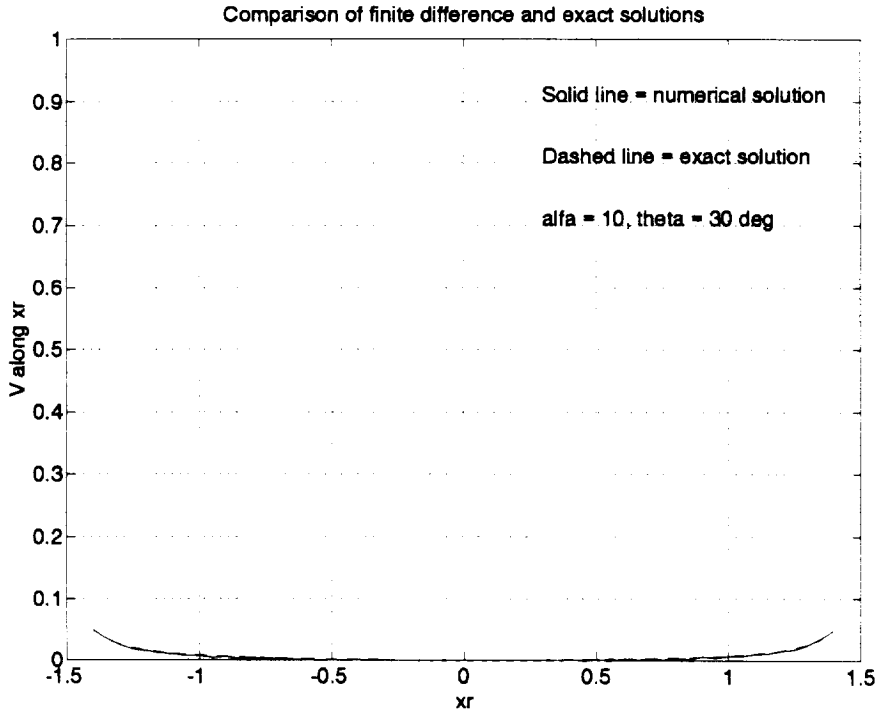


Figure 3-10: Exact vs. Finite Difference Solution to Zubov's P.D.E. — φ small

for each state $\mathbf{x}$. To implement the control law we can:

- compute the feedback gains online as the system evolves along a state trajectory. For this case we have found the following method to be effective:

  1. Assume we have an "old" solution to the Ricatti equation, say P_{old}. As the closed loop system progresses along a trajectory keep using the feedback gain:

  $$K(\mathbf{x}) = -R^{-1}B^T(\mathbf{x})P_{old} \tag{3.210}$$

  and "continuously" re-compute the residual Δ, where:

  $$\Delta(\mathbf{x}) = Q + P_{old}A(\mathbf{x}) + A^T(\mathbf{x})P_{old} - P_{old}B(\mathbf{x})R^{-1}B^T(\mathbf{x})P_{old} \tag{3.211}$$

  2. When

  $$\frac{\|\Delta(\mathbf{x})\|}{\|P_{old}\|} \geq \delta_R \tag{3.212}$$

  for some predefined $\delta_R > 0$ recompute the solution to the Ricatti Equation and update P_{old} with the newly computed value of $P(\mathbf{x})$.

- Precompute the feedback gains. Such an approach is examined in [62] where the feedback gains $K(\mathbf{x})$ are computed and then approximated as a sum of some basis functions, viz:

  $$K_{i,j}(\mathbf{x}) = \sum_{k=1}^{p} a_{i,j,k}\mathbf{f}_k(\mathbf{x}) \tag{3.213}$$

  Using basis functions to approximate the feedback gains has the advantage that it reduces the storage requirements.

Several methods exist to numerically solve the steady state Ricatti Equation. The methods can be classified as:

- Non-Iterative methods, such as eigenvector [39] and Schur Vector Methods [38].
- Iterative methods, such as Kleinman's-method [29].

We will summarize some of these methods below and introduce a new iterative method which we will refer to as Collar and Jahn's method. In each case we will assess the computational burden of the method by counting the number of "FLOPS" (floating point operations) required to solve the Ricatti equation. For comparison purposes we will count a scalar add and multiply,:

$$a + (b * c) \tag{3.214}$$

as one FLOP, and count FLOPS for other numerical procedures according to the following table (see e.g. [50], [38], [5]).

Operation	**Operation Count (FLOPS)**
Scalar add and multiply $a + (b * c)$	1
Matrix addition:$A + B \quad A, B \in R^{n \times n}$	n^2
Vector dotproduct: $\mathbf{x}^T\mathbf{y} \quad \mathbf{x}, \mathbf{y} \in R^{n \times 1}$	n
Matrix product: $A * B \quad A, B \in R^{n \times n}$	n^3
LU Decomposition: $A = LU$	$\frac{1}{3}n^3 + O(n^2)$
Solve: $AX = Y$ for X with $A, X, Y \in R^{n \times n}$	$\frac{4}{3}n^3 + O(n^2)$
Eigenvector Decomposition: $A = T\Lambda T^{-1}$	$8n^3 + O(n^2)$
Schur-vector Decomposition: $A = USU^T$	$8n^3 + O(n^2)$
Solve Lyapunov eqn. $PA + A^TP = Q \quad A, P, Q \in R^{n \times n}$	$12n^3 + O(n^2)$

3.5.1 Eigenvector Method

The procedure we refer to as the eigenvector method is based on the Mcfarlane-Potter-Fath algorithm [39]. The procedure for solving (3.209) is, (supressing the dependence on $\mathbf{x}$ for notational convenience):

C-2

1. Form the Hamiltonian Matrix:

$$\mathcal{H} = \begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix} \tag{3.215}$$

The number of operations in this step is essentially the work involved in forming $BR^{-1}B^T$.

2. Do an eigenvector de-composition of $\mathcal{H}$. If $\{A, B, C\}$ is stabilizable and detectable and the eigenvalues of $\mathcal{H}$ are distinct and we appropriately reorder the eigenvectors we get:

$$\mathcal{H}T = T\bar{\Lambda} \tag{3.216}$$

$$= \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} -\Lambda & 0 \\ 0 & \Lambda \end{bmatrix} \tag{3.217}$$

where T is a matrix containing the eigenvectors of $\mathcal{H}$ and $-\bar{\Lambda}$ is an $n \times n$ diagonal matrix containing the stable eigenvalues of $\mathcal{H}$.

3. The positive definite solution to the Ricatti equation is then given by:

$$P = T_{21}T_{11}^{-1} \tag{3.218}$$

4. Finally we compute the feedback gain:

$$K(\mathbf{x}) = R^{-1}B^T P \tag{3.219}$$

The following table sumarizes the operation counts for this method, where we have assumed that $B \in R^{n \times n}$:

Step	Operations count
Form $\mathcal{H}$:	
Form: $R^{-1}B^T$	$\frac{4}{3}n^3$
Multiply: $B * (R^{-1}B^T)$	n^3
Eigenvector decomposition of $\mathcal{H}$	$8(2n)^3$
Form $P = V_{21}V_{11}^{-1}$	$\frac{4}{3}n^3$
Compute feedback gain K:	
Multiply: $-(R^{-1}B^T) * P$	n^3
Note: $R^{-1}B^T$ is available from step 1	
TOTAL	$\approx 69n^3$

3.5.2 Schur-vector Method

The procedure we refer to as the Schur-vector method was developed by Laub [38] and differs from the Eigenvector method in that it uses the numerically more reliable Schur-vector decomposition [18] in stead of an Eigenvector decompostion. The procedure for solving (3.209) is, (supressing the dependence on **x** for notational convenience):

1. Form the Hamiltonian Matrix:

$$\mathcal{H} = \begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix} \tag{3.220}$$

2. Do a Schur-vector de-composition of $\mathcal{H}$. If $\{A, B, C\}$ is stabilizable and detectable we get:

$$\mathcal{H}U = US \tag{3.221}$$

$$= \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix} \tag{3.222}$$

where S is an upper triangular matrix with the eigenvalues of $\mathcal{H}$ appearing on its diagonal. It is furthermore assumed that the Schur-vectors have been re-ordered so that the stable eigenvalues of $\mathcal{H}$ appear on the diagonal of S_{11} while the unstable eigenvalues appear on the diagonal of S_{22}.

3. The positive definite solution to the Ricatti equation is then given by:

$$P = U_{21}U_{11}^{-1} \tag{3.223}$$

4. Finally we compute the feedback gain:

$$K(\mathbf{x}) = -R^{-1}B^T P \tag{3.224}$$

The following table sumarizes the operation counts for this method:

Step	**Operations count**
Form $\mathcal{H}$:	
Form: $R^{-1}B^T$	$\frac{4}{3}n^3$
Multiply: $B * (R^{-1}B^T)$	n^3
Schur-vector decomposition of $\mathcal{H}$	$8(2n)^3$
Form $P = S_{21}S_{11}^{-1}$	$\frac{4}{3}n^3$
Compute feedback gain K:	
Multiply: $-(R^{-1}B^T) * P$	n^3
Note: $R^{-1}B^T$ is available from step 1	
TOTAL	$\approx 69n^3$

3.5.3 Kleinman's Method

Kleinman's method [29] is an iterative procudure for solving the Ricatti equation. It uses the following algorithm assuming that a starting guess of P, say P_0 is available:

For $i = 0, 1 \ldots$

1. Form: $K_i = -R^{-1}B^T P_i$

2. Form: $\bar{A}_i = A - BK_i$

3. Solve the Matrix Lyapunov equation:

$$\bar{A}_i^T P_{i+1} + P_{i+1}\bar{A}_i = -(K_i^T R K_i + Q) \tag{3.225}$$

   for P_{i+1}

4. If $P_{i+1} = P_i$ we have a solution to (3.209). Otherwise iterate this process by setting $P_i \leftarrow P_{i+1}$ and starting again at step 1. Note that at the completion we already have the feedback gain available from step 1.

This iterative process should be natural to implement for our problem since we can assume that we have a previous value of $P(\mathbf{x})$ computed for a nearby state $\mathbf{x}$, which we can use to start the iterations. The following table gives an operation count for one iteration:

Step	**Operations count**
Form K_i:	
Multiply: $B^T * P_i$	$\approx n^3$
Form: $R^{-1}B^T P_i = K_i$	$\frac{4}{3}n^3$
Form $\bar{A}_k$:	
Multiply: $B * K_i$	n^3
Subtract: $A - (BK_i)$	n^2
Form $K_i^T R K_i + Q$ (Note: $K_i^T R K_i = (P_i B) * (R^{-1}B^T P_i)$, so use factors from first step)	
Multiply: $P_i B * (R^{-1}B^T P_i)$	n^3
Add: $K_i^T R K_i + Q$	n^2
Solve the Lyapunov Equation:	$12n^3$
TOTAL for k iterates	$\approx k * 16n^3$

3.5.4 Collar and Jahn's Method

The method we refer to as Collar and Jahn's method, is based on the Mcfarlane-Potter-Fath algorithm, but uses an iterative method to solve for the eigenstructure of the Hamiltonian matrix, in stead of using the more standard Hessenberg-QR-method (see e.g. [18]). The steps for Collar and Jahn's method are the same as for the eigenvector method outlined above except that the eigenstructure of $\mathcal{H}$ is found as follows [11]:

1. Assume that we have an iterate for the eigenvectors and eigenvalues of $\mathcal{H}$, say V_k and D_k. Find Δ_k and D_{k+1} from:

$$D_{k+1} = \text{diag}\left[V_k^{-1}\mathcal{H}V_k\right] \tag{3.226}$$

$$\Delta_k = V_k^{-1}\mathcal{H}V_k - D_{k+1} \tag{3.227}$$

   i.e. D_{k+1} consists of the diagonal elements of $V_k^{-1}\mathcal{H}V_k$, and Δ_k consists of the off-diagonal elements of the same matrix.

2. Solve for E_k from:

$$E_k D_{k+1} - D_{k+1}E_k = \Delta_k \tag{3.228}$$

   Since D_{k+1} is diagonal, the solution is simply given by:

$$[E_k]_{ij} = \frac{[\Delta_k]_{ij}}{[D_{k+1}]_{ii} + [D_{k+1}]_{jj}} \tag{3.229}$$

3. Form the updated estimate of V:

$$V_{k+1} = V_k(I + E_k) \tag{3.230}$$

4. Repeat the process until Δ_k is small enough.

The operation count for one iterate is given by:

Step	Operations count
Form $\mathcal{H}$:	
Form: $R^{-1}B^T$	$\frac{4}{3}n^3$
Multiply $B * (R^{-1}B^T)$	n^3
Update Eigenvectors of $\mathcal{H}$	
Form: $V_k^{-1}\mathcal{H}$	$\frac{4}{3}(2n^3)$
Multiply: $(V_k^{-1}\mathcal{H}) * V_k$	$(2n)^3$
Solve for Δ_k	$(2n)^2$
Form: $V_{k+1} = V_k * (I + E_k)$	$(2n)^3$
Form $P = V_{21}V_{11}^{-1}$	$\frac{4}{3}n^3$
Compute the feedback gain K:	
Multiply $-(R^{-1}B^T) * P$	n^3
TOTAL for k iterates	$\approx k * 31n^3$

3.5.5 Recommendation

Based on the operation counts we have found above we generally prefer the Schur-vector method for on-line computation of the feedback gains, since:

- it is competitive in terms of computational load (assuming the iterative methods require 3 to 5 iterations per solution)
- it is numerically more reliable than the eigenvector method
- it has a more predictable execution time — it does not have the open ended iterative nature of the other two methods.

3.6 Simulation Results

In this section we will apply the continuous Ricatti design method to a nonlinear system. We will see that the method generally works better that designs based on the linearized dynamics but that at some point it becomes unstable. An approach to improve the stability properties will be discussed in Chapter 4 where we will use an optimal control approach which utilizes knowledge about the future behavior of $A(\mathbf{x}), B(\mathbf{x})$.

3.6.1 System Dynamics

The system we use for the simulations will have dynamics related to that of a one-link flexible manipulator. We will change the dynamics of a one link arm by simplifying the equations of motion on the one hand, but at the same time introduce parameters which can be used to accentuate the main nonlinearity in the problem.

To derive the equations of motion for the one link flexible arm of Figure 3-11 we use the assumed modes approach discussed in Appendix A. For this example we assumed a single sinusoid mode shape so that the displacement of each "slice" of the arm relative to the line joining the root and the tip (see Figure 3-11) is given by:

$$v(\zeta, q) = \sin\left(\frac{\pi\zeta}{L}\right) \tag{3.231}$$

where:

$$L = \text{length of the link} \tag{3.232}$$

$$\zeta = \text{"location" of the slice} \tag{3.233}$$

In what follows we shall refer to θ in Figure 3-11 as the *gross motion angle* and to q as the *flex mode deflection.*

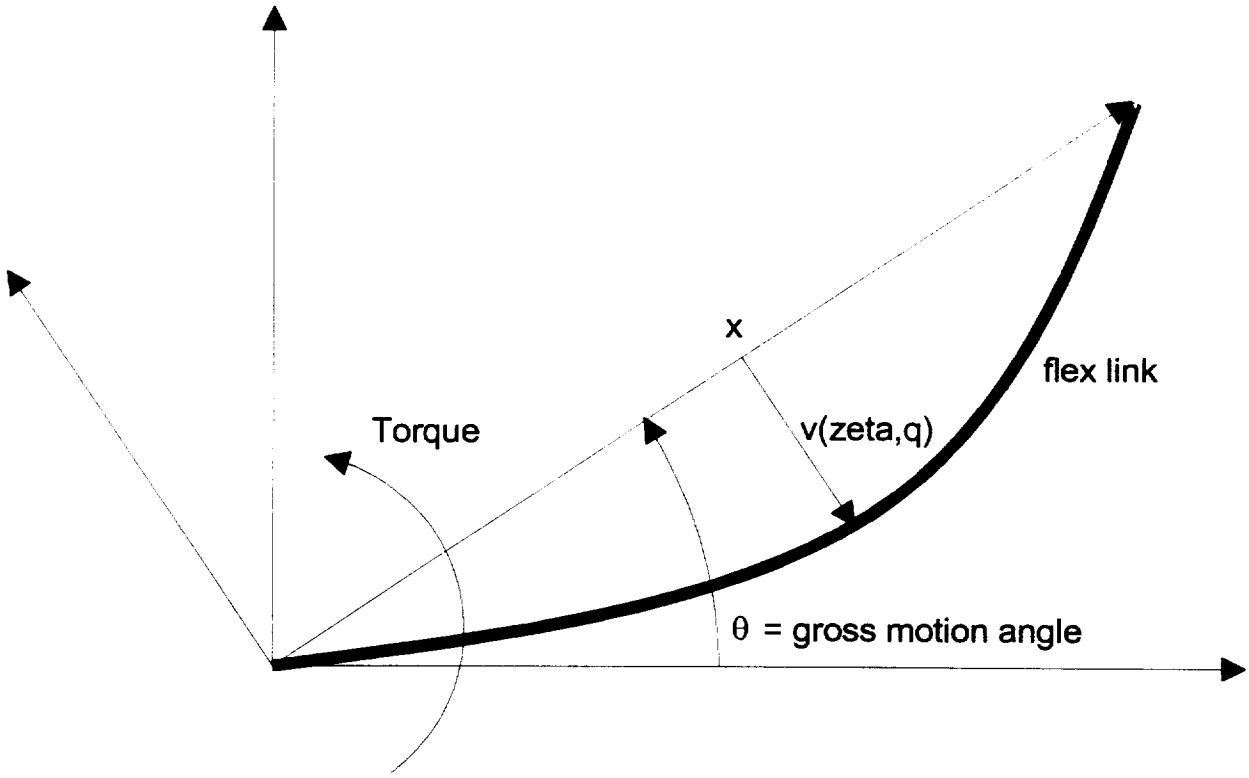


Figure 3-11: One Link Arm

The resulting equations of motion can be put in quasilinear form. We get:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})u \tag{3.234}$$

with:

$$\mathbf{x}^T = \begin{bmatrix} \theta & q & \dot{\theta} & \dot{q} \end{bmatrix} \tag{3.235}$$

$$A(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & a_{32} & a_{33} & 0 \\ 0 & a_{42} & a_{43} & 0 \end{bmatrix} \tag{3.236}$$

$$B(\mathbf{x}) = \begin{bmatrix} 0 \\ 0 \\ \frac{-8.905}{\gamma(q)} \\ -0.0349\frac{4.636\times10^3+29.61q^2}{\gamma(q)} \end{bmatrix} \tag{3.237}$$

where

$$a_{32} = \frac{\left(-0.218\times10^7+18.85(740.2\dot{\theta}^2)\right)}{\gamma(q)} \tag{3.238}$$

$$a_{33} = \frac{-18.85q(3.142\dot{q})}{\gamma(q)} \tag{3.239}$$

$$a_{42} = \frac{-8.5\times10^4(3.6\times10^2+3q^2)+1.733\times10^2(1.132\dot{\theta}^2)}{\gamma(q)} \tag{3.240}$$

$$a_{43} = \frac{-1.733\times10^2q(1.61\times10^2\dot{q}-9.425\dot{\theta}q^2)}{\gamma(q)} \tag{3.241}$$

$$\gamma(q) = 1.157\times10^5-2.961\times10^2q^2 \tag{3.242}$$

and u is the control input. We used parameters similar to the space shuttle R.M.S., viz:

$$L \text{ (length)} = 13.44m \tag{3.243}$$

$$\rho_A \text{ (density per unit length)} = 55.16kg/m \tag{3.244}$$

$$EI \text{ (stiffness)} = 1.38\times10^11*2.08\times10^{-5}Nm^2 \tag{3.245}$$

If we neglect terms like $q^2, q\dot{q}, q^3\dot{\theta}$ and round some of the constants $A(\mathbf{x}), B(\mathbf{x})$ simplify too:

$$A(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\alpha_{11}+\alpha_{12}\dot{\theta}^2 & 0 & 0 \\ 0 & -\alpha_{21}+\alpha_{22}\dot{\theta}^2 & 0 & 0 \end{bmatrix} \tag{3.246}$$

$$B(\mathbf{x}) = \begin{bmatrix} 0 \\ 0 \\ -\beta_1 \\ -\beta_2 \end{bmatrix} \tag{3.247}$$

where for the nominal case:

$$\alpha_{11} = 20 \tag{3.248}$$

$$\alpha_{12} = 0.12 \tag{3.249}$$

$$\alpha_{21} = -267 \tag{3.250}$$

$$\alpha_{22} = 1.7 \tag{3.251}$$

$$\beta_1 = 8 \times 10^{-5} \tag{3.252}$$

$$\beta_2 = 1.4 \times 10^{-3} \tag{3.253}$$

To find a feedback law we used the approach discussed in Section 3.2.3. Our goal was to cause the gross motion angle θ and angular rate $\dot{\theta}$ to remain similar to that of a reference model while at the same time keeping the flex mode deflection small. We used the cost function and augmented dynamics of equations (3.47), (3.48), (3.53).

The reference model we used is:

$$\dot{\mathbf{x}}_d = A_d \mathbf{x}_d \tag{3.254}$$

$$= \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} \begin{bmatrix} \theta_d \\ \dot{\theta}_d \end{bmatrix} \tag{3.255}$$

where:

$$\omega_n = 10 \tag{3.256}$$

$$\zeta = 1 \tag{3.257}$$

For this example (which is intended to be stressful to the continuous Ricatti design method) we used the following parameters in the cost function (see equation 3.53):

$$\mathbf{e}^T = \begin{bmatrix} (\theta - \theta_d) & (\dot{\theta} - \dot{\theta}_d) \end{bmatrix} \tag{3.258}$$

$$\mathbf{z}^T = \begin{bmatrix} \theta & q & \dot{\theta} & \dot{q} & \theta_d & \dot{\theta}_d \end{bmatrix} \tag{3.259}$$

and:

$$Q_y = \text{diag}\begin{bmatrix} 1 \times 10^7 & 1 \times 10^5 \end{bmatrix} \tag{3.260}$$

$$Q_z = \text{diag}\begin{bmatrix} 0 & 1 \times 10^5 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.261}$$

$$\rho = 1 \times 10^{-4} \tag{3.262}$$

Case 1

For the first case we used the cost function etc. as specified above and accentuated the nonlinearity of the system by using the following values for α_{ij} in equations (3.246), (3.247):

$$\alpha_{11} = 20 \tag{3.263}$$

$$\alpha_{12} = 0.12 \tag{3.264}$$

$$\alpha_{21} = -26 \tag{3.265}$$

$$\alpha_{22} = 50 \tag{3.266}$$

$$\beta_1 = 8 \times 10^{-5} \tag{3.267}$$

$$\beta_2 = 1.4 \times 10^{-3} \tag{3.268}$$

The desired response is a 90° re-orientation of the "arm" in about $0.5s$ which is a strenuous manever for an "arm" of this size. Figure 3-12 shows the stable response of the closed loop system when using the continuous Ricatti design feedback, while Figure 3-13 shows that the closed loop response for the feedback law based on the

linearized dynamics is unstable.

Case 2

For this case we further accentuated the nonlinearity by using:

$$\alpha_{11} = 20 \tag{3.269}$$

$$\alpha_{12} = 0.12 \tag{3.270}$$

$$\alpha_{21} = -26 \tag{3.271}$$

$$\alpha_{22} = 170 \tag{3.272}$$

$$\beta_1 = 8 \times 10^{-5} \tag{3.273}$$

$$\beta_2 = 1.4 \times 10^{-3} \tag{3.274}$$

At this point we see from Figure 3-14 that the continuous Ricatti design feedback law results in unstable behavior. We will revisit this case in Chapter 4 where we will discuss an approach to improve the stability properties by using an optimal control approach which utilizes knowledge about the future behavior of $A(\mathbf{x})$ and $B(\mathbf{x})$.

3.7 Summary

In this Chapter we explored a control design method that exploits the quasilinear form of dynamics for nonlinear systems. The method can be applied to a large class of nonlinear systems and provides the designer with convenient design parameters to modify the closed loop system behavior. It has been our experience (see also Chapter 5 and [62] for further examples) that the method resulted in well behaved closed loop systems if we appropriately adjusted the design parameters. However only local stability can be guaranteed. To analyze the "non-local" stability properties we showed the connection between the continuous Ricatti design method and receding

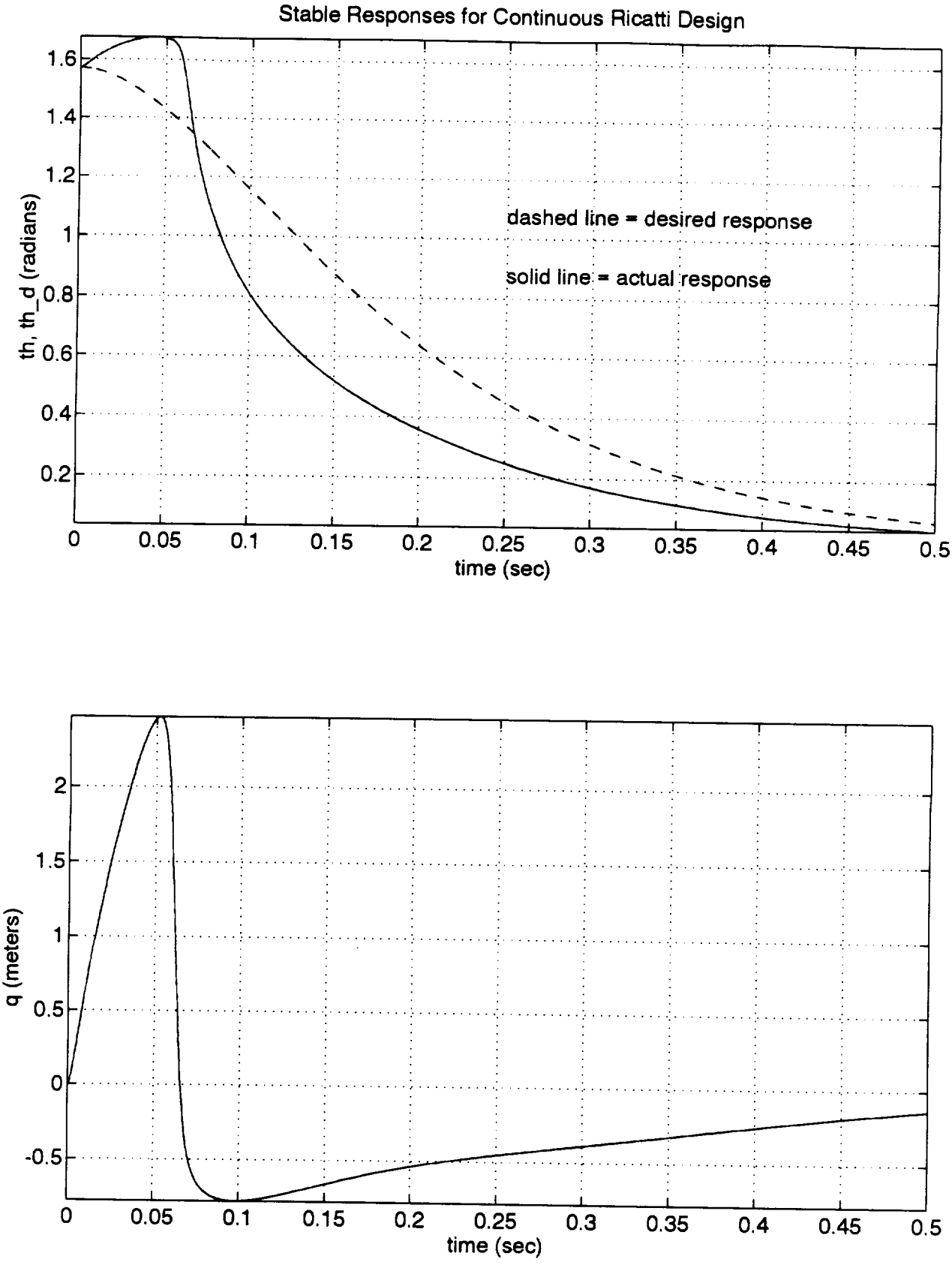


Figure 3-12: Stable Response — Continuous Ricatti Design

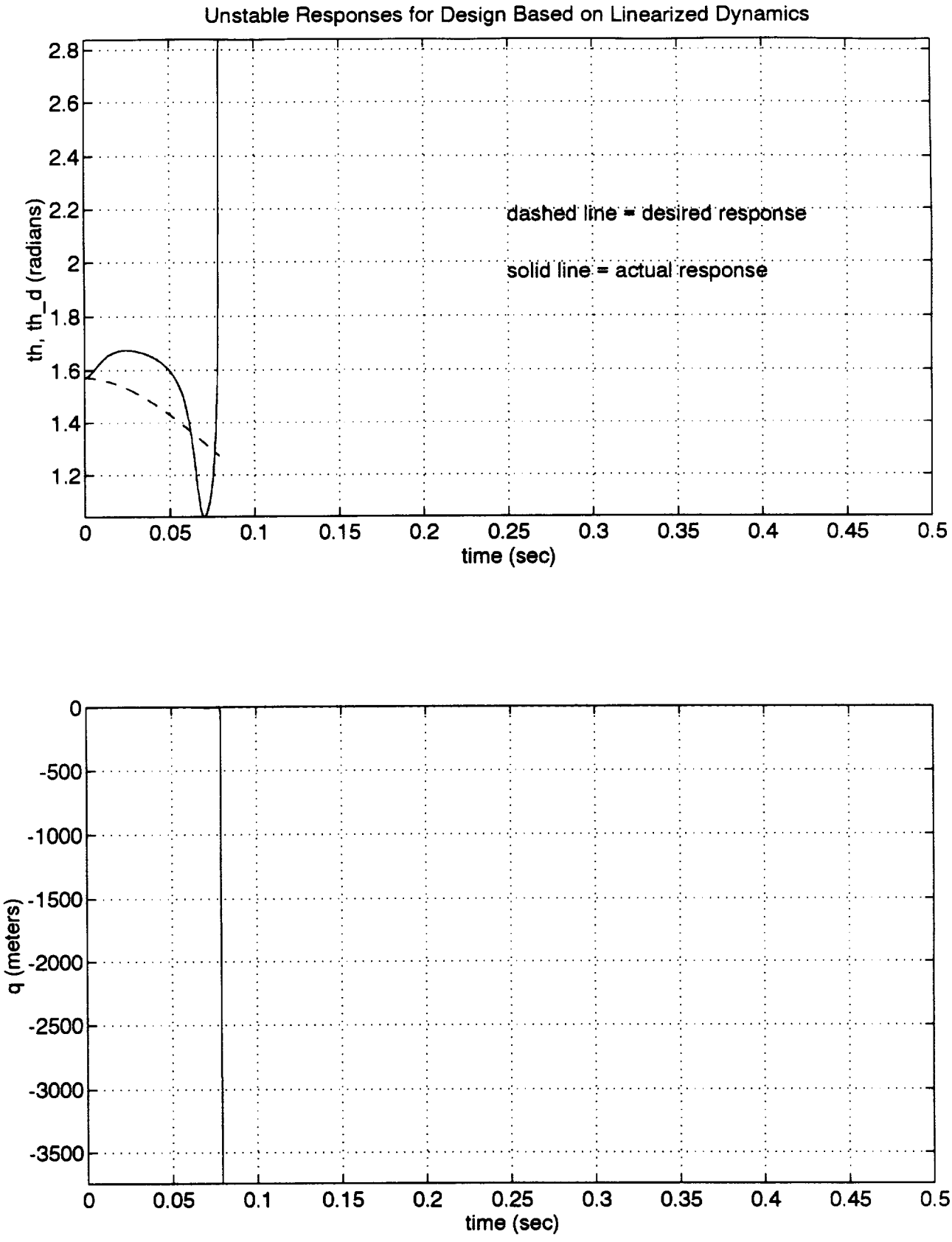


Figure 3-13: Unstable Response — Linearized Design

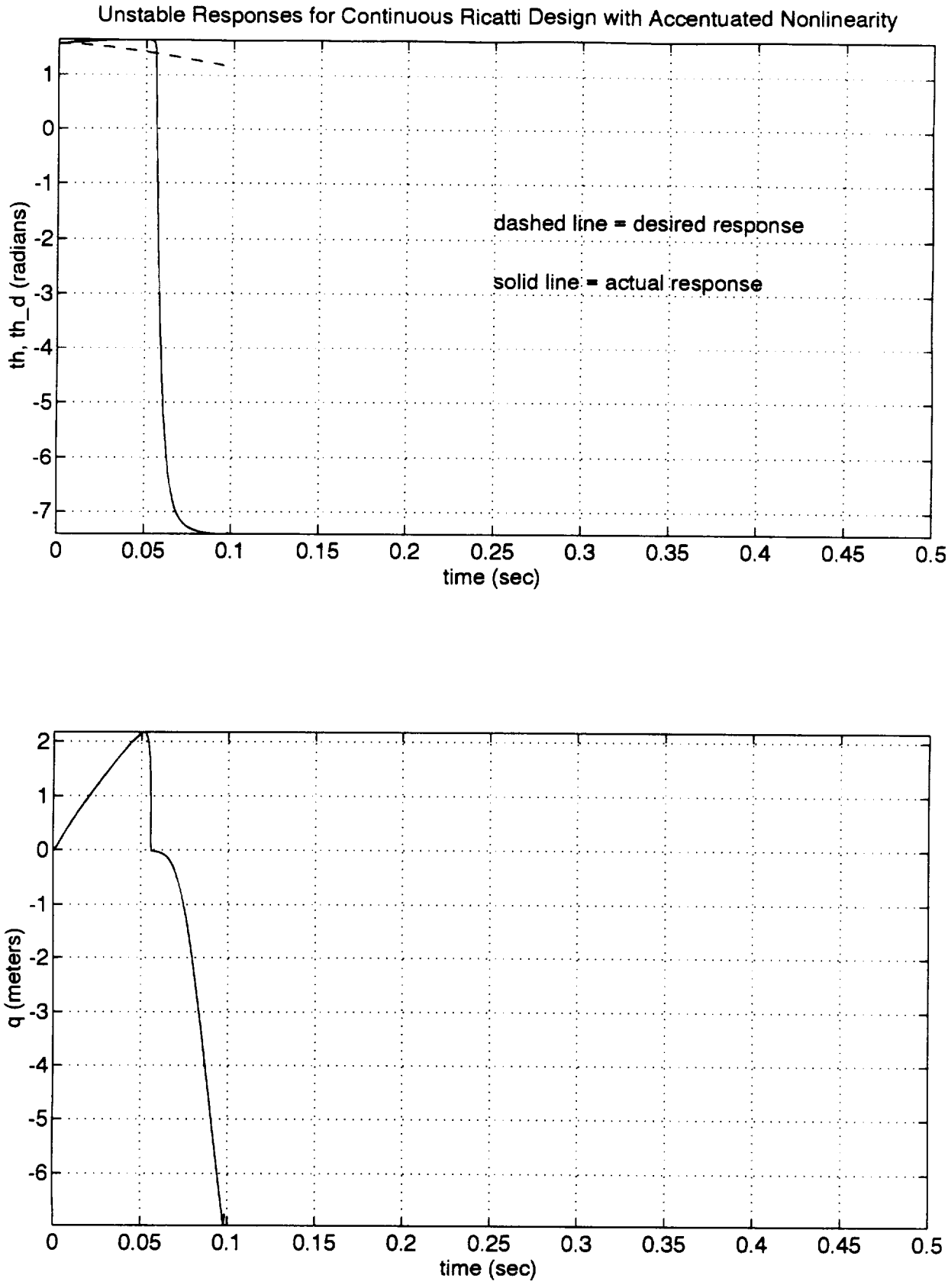


Figure 3-14: Unstable Response — Continuous Ricatti Design

horizon control. We also showed how Zubov's method could be used to numerically determine the stability domain of the closed loop system.

In the next chapter we will explore an optimal control methodology that further exploits the quasilinear form of the dynamics and can be used to improve closed loop stability.

Chapter 4

Controllers with Look-Ahead

4.1 Introduction

In this chapter we further explore and exploit the similarity between linear time-varying systems and nonlinear systems in quasilinear form. The main difference between the methods presented here and the continuous Ricatti design method of the previous chapter is that the new control methods take into account "knowledge" about the future behavior of $A(\mathbf{x})$ and $B(\mathbf{x})$. Such controllers will be refered to as "controllers with look-ahead".

Sections 4.2, 4.3 and 4.4 examine control strategies for general (not necessarily feedback linearizable) nonlinear systems which will result in stable closed loop systems. The methods are all rooted in optimal control theory and require one to solve optimization problems of the form:

Given a system:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \quad \mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.1}$$

$$\mathbf{y} = \mathbf{h}(\mathbf{x}) \tag{4.2}$$

Find the control input $\mathbf{u}$ *that will minimize the cost function:*

$$J = \varphi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} \mathcal{L}(\mathbf{x}, \mathbf{u}, \tau)\mathrm{d}\tau, \quad \mathcal{L}(\mathbf{x}, \mathbf{u}, \tau) \geq 0 \tag{4.3}$$

Note that since $\mathcal{L}$ is explicitly a function of time, the optimization problem as defined above is sufficiently general to include tracking problems. For example assume that the plant output $\mathbf{y}$ is required to track some reference trajectory, say $\mathbf{y}_d(t)$, then we can use:

$$\mathcal{L}(\mathbf{x}, \mathbf{u}, t) = \mathcal{L}(\mathbf{x}, \mathbf{u}, \mathbf{y}_d(t)) \tag{4.4}$$

Once the general theory for the methods is in place, we develop an approach to (approximately) solve the required optimal control problems by using the similarity between quasilinear systems and linear time varying systems (see Section 4.5).

Assumption 4.1.1 *We will assume that our systems are sufficiently well behaved so that if we apply the optimal control for the problem defined by equations (4.1), (4.2), (4.3), then* $\mathcal{L}(\mathbf{x}, \mathbf{u}, \tau)$ *will remain bounded over the interval* $[t_0, t_f]$ *and* $\varphi(\mathbf{x}(t_f))$ *will also be bounded.*

Assumption 4.1.1 is needed to ensure that $\mathcal{L}(\mathbf{x}, \mathbf{u}, t)$ does not, for instance, behave like $\left(\frac{1}{t-a}\right)^{\frac{2}{3}}$ which becomes unbounded at $t = a$ but remains integrable across the singularity. Note that Assumption 4.1.1 does not in itself guarantee that the closed loop system will be stable, since some of the system states may well be unstable but not affect $\mathcal{L}$.

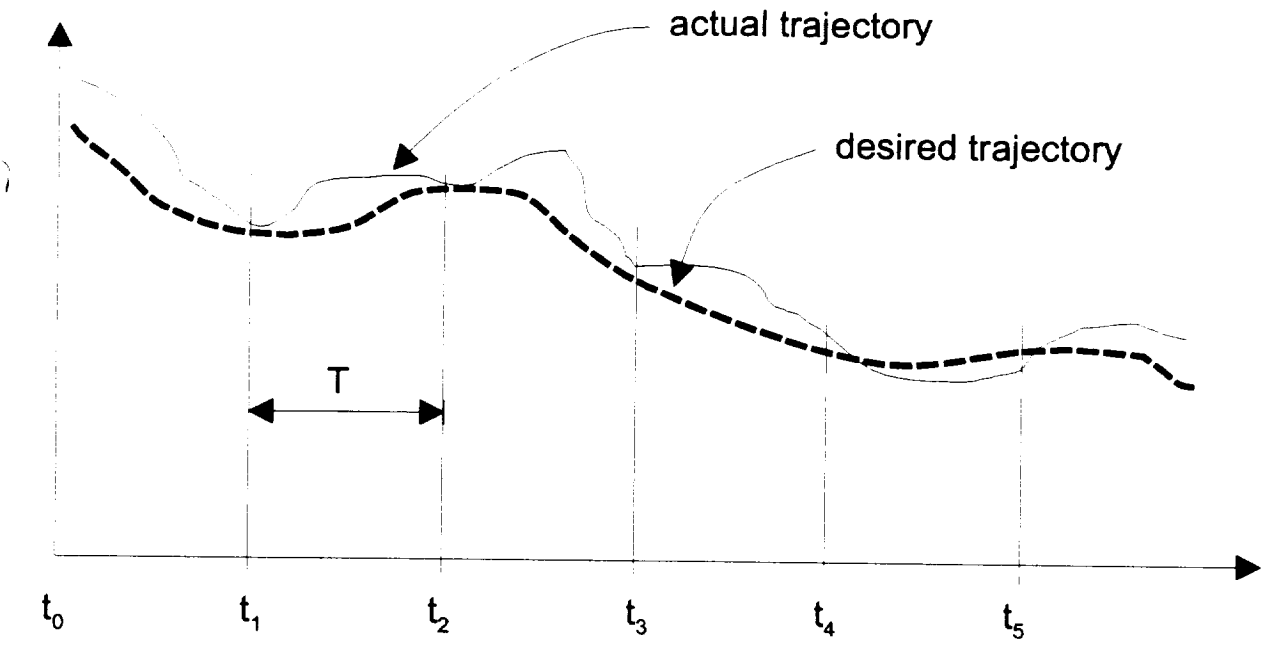


Figure 4-1: Time Intervals for Short Term Tracker

4.2 Short Term Optimal Tracking

This section examines an approach to control nonlinear systems by solving a sequence of short term optimization problems. The goal of the method is to cause the output of the system (4.1),(4.2) to track some desired trajectory, say $\mathbf{y}_d(t)$ and at the same time ensure that all the state variables remain bounded (see also Section 3.2.3). It is furthermore assumed that at any time t the reference trajectory is only known over a limited interval (horizon), say $[t, t+T]$.

To achieve the tracking/regulation goal, given the limited knowledge of the desired trajectory, the control law is found by solving a sequence of optimization problems each defined for a time interval $[t_i, t_i + T]$. $i = 0, 1 \ldots$ (see also Figure 4-1). Lemma 4.2.1 shows that under certain conditions this approach will lead to stable closed loop systems. Before showing the desired stability result we establish the following well known fact:

Fact 4.2.1 *Assume that the nonlinear system:*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \tag{4.5}$$

is completely controllable. Consider the cost function:

$$J = \mu \mathbf{x}^T(t_f) H \mathbf{x}(t_f) + \int_{t_0}^{t_f} \mathcal{L}(\mathbf{x}, \mathbf{u}, \tau) \mathrm{d}\tau \tag{4.6}$$

where $\mathcal{L}(\mathbf{x}, \mathbf{u}, \tau) \geq 0$ *and* H *is positive semi-definite. Let* $\bar{\mathbf{u}}_\mu[t_0, t_f]$ *be the control time history which minimizes* J *for a specific value of* μ*, and let* $\bar{\mathbf{x}}_\mu[t_0, t_f]$ *be the associated state trajectory. Then if we apply the optimal input* $\bar{\mathbf{u}}_\mu$ *to the system,* $\bar{\mathbf{x}}_\mu^T(t_f) H \bar{\mathbf{x}}_\mu(t_f)$ *will remain finite. Furthermore as* $\mu \to \infty$ *we will have* $\bar{\mathbf{x}}_\mu(t_f) H \bar{\mathbf{x}}_\mu(t_f) \to 0$.

Proof:

Since (4.5) is completely controllable there exists some control history, say $\tilde{\mathbf{u}}[t_0, t_f]$, which drives the system from its initial condition to a state where $\mathbf{x}^T(t_f) H \mathbf{x}(t_f) = 0$. Let $\tilde{\mathbf{x}}$ be the associated state trajectory and

$$\begin{aligned} \tilde{J} &= \tilde{\mathbf{x}}^T H \tilde{\mathbf{x}} + \int_{t_0}^{t_f} L(\tilde{\mathbf{x}}, \tilde{\mathbf{u}}, \tau) \mathrm{d}\tau & (4.7) \\ &= \int_{t_0}^{t_f} L(\tilde{\mathbf{x}}, \tilde{\mathbf{u}}, \tau) \mathrm{d}\tau & (4.8) \end{aligned}$$

be the cost associated with using $\tilde{\mathbf{u}}$ as the control. Since $\bar{\mathbf{u}}_\mu$ is optimal it follows that:

$$0 \leq \mu \, \bar{\mathbf{x}}_\mu^T H \bar{\mathbf{x}}_\mu \Big|_{t_f} + \int_{t_0}^{t_f} \mathcal{L}(\bar{\mathbf{x}}_\mu, \bar{\mathbf{u}}_\mu, \tau) \mathrm{d}\tau \leq \tilde{J} \tag{4.9}$$

Since $\mathcal{L}(\mathbf{x}, \mathbf{u}, \tau) \geq 0$ we then have:

$$\begin{aligned} \mu \, \bar{\mathbf{x}}_\mu^T H \bar{\mathbf{x}}_\mu \Big|_{t_f} &\leq \tilde{J} & (4.10) \\ \Rightarrow \bar{\mathbf{x}}_\mu^T H \bar{\mathbf{x}}_\mu \Big|_{t_f} &\leq \frac{\tilde{J}}{\mu} & (4.11) \end{aligned}$$

Since $\tilde{J}$ is finite $\bar{\mathbf{x}}_\mu H \bar{\mathbf{x}}_\mu$ will be finite. Furthermore since $\tilde{J}$ does not depend on μ we

can make the left hand side of (4.11) arbitrarily small by making μ large enough.

———*Q.E.D.*

With this fact in hand it can be shown that under certain conditions the strategy outlined above will result in stable closed loop systems (in a bounded input-bounded output sense). In order to simplify the statement of the proof of stability for this method we assume that a state transformation has been applied to the system so that the state vector explicitly contains the plant output, i.e:

$$\mathbf{x} = \begin{bmatrix} \mathbf{y} \\ \mathbf{x}_r \end{bmatrix} \tag{4.12}$$

Lemma 4.2.1 *Consider the nonlinear system (4.5). Assume that the system is reachable for any time interval $> T_r$, and that the reference trajectory $\mathbf{y}_d$ is bounded. Divide the time axis into a sequence of time intervals $[t_i, t_i + T], i = 0, 1, \ldots$ with $T > T_r$ (see e.g. Figure 4-1) , and consider a sequence of optimization problems, one for each time interval, where the cost function for each interval is:*

$$J_i = \mu \left(\mathbf{e}^T H_y \mathbf{e} + \mathbf{x}_r^T H_x \mathbf{x}_r\right)\Big|_{t_f} + \int_{t_i}^{t_f} (\mathbf{e}^T Q_y \mathbf{e} + \mathbf{x}_r Q_x \mathbf{x}_r + \mathbf{u}^T R \mathbf{u}) \mathrm{d}\tau \tag{4.13}$$

where $\mathbf{e} \stackrel{\text{def}}{=} \mathbf{y} - \mathbf{y}_d$ and Q_y, Q_x, H_y, H_x are all positive definite. Let $\bar{\mathbf{u}}_i[t_i, t_i + T]$ be the control input which is optimal for the i'th problem. If Assumption 4.1.1 holds, and μ is large enough, and we then apply $\bar{\mathbf{u}}_i[t_i, t_i + T]$ on each interval, the closed loop system will be stable.

Proof:

Consider an arbitrary interval $[t_i, t_i + T]$. Assume that the state of the system at the beginning of the interval, i.e. $\mathbf{x}(t_i)$, is bounded. Then Fact 4.2.1 implies that, if $\bar{\mathbf{u}}[t_i, t_i + T]$ is applied over the interval $[t_i, t_f]$, then $\mathbf{e}^T H_y \mathbf{e} + \mathbf{x}_r^T H_x \mathbf{x}_r$ will be arbitrarily small, provided that μ is large enough. Since H_y and H_x are positive definite, this implies that $\mathbf{e}(t_f)$ and $\mathbf{x}_r(t_f)$ will be finite, and subsequently since $\mathbf{y}_d$ is bounded,

$\mathbf{x}(t_f)$ must be bounded. Furthermore since Q_y, Q_x are positive definite, Assumption 4.1.1 ensures that $\mathbf{x}$ will remain bounded. At this point we have established that: if the system state is bounded at the beginning of an interval, it will remain bounded throughout the interval up to the beginning of the next interval. Assuming that the state of the system at t_0 is bounded, it follows that the system response is bounded for all intervals.

Q.E.D.

The result above shows that by solving a sequence of short term optimal control problems we can obtain stable closed loop systems. Unfortunately, in practice, this approach has limited value since the control history generally is discontinuous at the boundary of each interval and the control law is thus likely to excite unmodeled dynamics which may lead to instability. (See also Example 4.6.1).

4.3 Receding Horizon Control

The previous section examined a stable control law which required that a sequence of short term optimization problems had to be solved. The method had the drawback that it would lead to discontinuous control histories. To remedy the problem of the discontinuous control we examine another control methodology, i.e. receding horizon control (see e.g. [36], [41]).

The receding horizon control strategy is:

1. At the current state, say $\mathbf{x}(t)$ solve an optimization problem of the type defined by equations (4.1), (4.2), (4.3), over a finite horizon $[t, t+T]$.

2. If $\bar{\mathbf{u}}[t, t+T]$ is the solution to the optimization problem over the full interval, apply only the initial control, i.e $\bar{\mathbf{u}}(t)$, at the initial state.

3. Repeat the process starting at step 1.

Fundamental results regarding the stability of this control law are available. Kwon and Pearson [36] show that for linear time varying systems:

$$\dot{\mathbf{x}} = A(t)\mathbf{x} + B(t)\mathbf{u} \tag{4.14}$$

with a cost function of the form:

$$J = \int_t^{t+T} (\mathbf{x}^T Q\mathbf{x} + \mathbf{u}^T R\mathbf{u}) \mathrm{d}\tau \tag{4.15}$$

and subject to the terminal constraint that $\mathbf{x}(t+T) \equiv \mathbf{0}$ the closed loop system will remain stable. Mayne and Michalska [41] expand this result to nonlinear systems. They show that if the receding horizon control law is applied to systems of the form (4.1) with cost function:

$$J = \int_t^{t+T} (\mathbf{x}^T Q\mathbf{x} + \mathbf{u}^T R\mathbf{u}) \mathrm{d}\tau \quad Q, R > 0 \tag{4.16}$$

and subject to the terminal constraint that $\mathbf{x}(t+T) \equiv \mathbf{0}$ the closed loop system will be stable. Note that in both the linear and nonlinear system cases the above-mentioned stability results cannot be applied as is to tracking problems, since for tracking problems $\mathbf{y}_d(t+T) \neq \mathbf{0}$ in general. Stability results for tracking problems appear in [34] for linear time invariant discrete time systems:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k) \tag{4.17}$$

$$\mathbf{y} = C\mathbf{x}(k) \tag{4.18}$$

and cost functions of the form:

$$J = \frac{1}{2}\left(\mathbf{e}^T(k+N)H\mathbf{e}(k+N)\right) + \frac{1}{2}\sum_{j=k}^{j=k+N}\left(\mathbf{e}^T(j)Q\mathbf{e}(j) + \mathbf{u}(j)^T R\mathbf{u}(j)\right) \tag{4.19}$$

where:

$$\mathbf{e}(j) \stackrel{\text{def}}{=} \mathbf{y}(j) - \mathbf{y}_d(j) \tag{4.20}$$

and it is assumed that the reference trajectory $\mathbf{y}_d(j)$ is available over the horizon $[k, k+N]$.

In what follows we extend the stability results mentioned above to the nonlinear receding horizon tracking problem (i.e. where $\mathbf{x}(t+T) \neq 0$). In the process we also develop an alternate method of proof for the nonlinear receding horizon "regulator" stability result by Mayne and Michalska (i.e. where $\mathbf{x}(t+T) \equiv 0$). Our results depend on the following Lemma:

Lemma 4.3.1 *Consider the nonlinear system (4.1) and cost function (4.3). Assume that the system is reachable and allow for the possibility of a terminal constraint of the form* $\psi(\mathbf{x}(t_f)) = \mathbf{0}$. *Let* $\bar{\mathbf{u}}[t_0, t_f]$, *and* $\bar{\mathbf{x}}[t_0, t_f]$ *denote the optimal control and state trajectories respectively and let:*

$$V(\mathbf{x}, t_0, t_f) = \int_{t_0}^{t_f} (\mathcal{L}(\bar{\mathbf{x}}(\tau), \bar{\mathbf{u}}(\tau), \tau) \mathrm{d}t \tag{4.21}$$

denote the actual cost incurred if we apply the optimal control $\bar{\mathbf{u}}$ *over the interval* $[t_0, t_f]$ *starting from state* $\mathbf{x}$ *at time* t_0. *Then:*

$$\frac{\mathrm{d}}{\mathrm{d}t} V(\mathbf{x}, t, t+T)\Big|_{\dot{\mathbf{x}}=\mathbf{f}(\mathbf{x},\bar{\mathbf{u}},t)} = -\mathcal{L}(\mathbf{x}, \bar{\mathbf{u}}, t) + \frac{\partial V(\mathbf{x}, t_0, t_f)}{\partial t_f}\Big|_{t_f=t+T} \tag{4.22}$$

Proof:

We have:

$$\begin{aligned} \frac{\mathrm{d}V(\mathbf{x}, t, t+T)}{\mathrm{d}t} &= \frac{\partial V(\mathbf{x}, t_0, t_f)}{\partial \mathbf{x}}\Big|_{\substack{t_0=t \\ t_f=t+T}} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} + \frac{\partial V(\mathbf{x}, t_0, t_f)}{\partial t_0}\Big|_{\substack{t_0=t \\ t_f=t+T}} \\ &\quad + \frac{\partial V(\mathbf{x}, t_0, t_f)}{\partial t_f}\Big|_{\substack{t_0=t \\ t_f=t+T}} \end{aligned} \tag{4.23}$$

$$= \left.\frac{\partial V(\mathbf{x},t_0,t_f)}{\partial \mathbf{x}}\right|_{\substack{t_0=t\\ t_f=t+T}} \mathbf{f}(\mathbf{x}(t),\mathbf{u}(t),t) + \left.\frac{\partial V(\mathbf{x},t_0,t_f)}{\partial t_0}\right|_{\substack{t_0=t\\ t_f=t+T}} + \left.\frac{\partial V(\mathbf{x},t_0,t_f)}{\partial t_f}\right|_{\substack{t_0=t\\ t_f=t+T}} \tag{4.24}$$

From the HJB-equation we know that:

$$-\frac{\partial V(\mathbf{x},t_0,t_f)}{\partial t_0} = \mathcal{L}(\mathbf{x},\bar{\mathbf{u}}(t_0),t_0) + \frac{\partial V(\mathbf{x},t_0,t_f)}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x},\bar{\mathbf{u}}(t_0),t_0) \tag{4.25}$$

Substituting (4.25),into (4.24) then gives us equation (4.22) as desired.

———*Q.E.D.*

We can find an expression for

$$\frac{\partial V(\mathbf{x},t_0,t_f)}{\partial t_f} \tag{4.26}$$

as follows:

Fact 4.3.1 *Consider an optimization problem where we have a system as in (4.1), a cost function:*

$$J = \varphi(\mathbf{x}(t_f),t_f) + \int_{t_0}^{t_f} \mathcal{L}(\mathbf{x},\mathbf{u},\tau)\mathrm{d}\tau \tag{4.27}$$

and a terminal constraint $\boldsymbol{\psi}(\mathbf{x}(t_f),t_f) = \mathbf{0}$. *Let* $V(\mathbf{x},t_0,t_f)$ *again be the actual cost incurred if we apply the optimal control. Then:*

$$\frac{\partial V(\mathbf{x},t_0,t_f)}{\partial t_f} = \left.\left(\frac{\partial \Phi(\mathbf{x},t_f)}{\partial \mathbf{x}}\dot{\mathbf{x}} + \frac{\partial \Phi(\mathbf{x},t_f)}{\partial t_f} + \mathcal{L}(\mathbf{x},\mathbf{u},t_f)\right)\right|_{\substack{t_f\\ \dot{\mathbf{x}}(t_f)}} \tag{4.28}$$

where:

$$\Phi(\mathbf{x},t_f) \stackrel{\text{def}}{=} \boldsymbol{\nu}^T\boldsymbol{\psi}(\mathbf{x}(t_f),t_f) + \varphi(\mathbf{x}(t_f),t_f) \tag{4.29}$$

and $\boldsymbol{\nu}$ *is a Lagrange multiplier.*

Proof:

Let $\bar{\mathbf{u}}[t_0,t_f]$ and $\bar{\mathbf{x}}[t_0,t_f]$ be the optimal control and state trajectory associated with

the optimization problem defined above, i.e. over $[t_0, t_f]$. Define the augmented cost function:

$$J_a \stackrel{\text{def}}{=} \boldsymbol{\nu}^T \boldsymbol{\psi}(\mathbf{x}(t_f), t_f) + \varphi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} \left(\mathcal{L}(\mathbf{x}, \mathbf{u}, \tau) + \boldsymbol{\lambda}^T(-\dot{\mathbf{x}} + \mathbf{f}(\mathbf{x}, \mathbf{u}, \tau)) \right) \quad (4.30)$$

$$\stackrel{\text{def}}{=} \Phi(\mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} \left(\mathcal{H}(\mathbf{x}, \mathbf{u}, \tau) - \boldsymbol{\lambda}^T \dot{\mathbf{x}} \right) \mathrm{d}\tau \quad (4.31)$$

where $\boldsymbol{\nu}, \boldsymbol{\lambda}$ are Lagrange multipliers and we have defined:

$$\mathcal{H}(\mathbf{x}, \mathbf{u}, \tau) \stackrel{\text{def}}{=} \mathcal{L}(\mathbf{x}, \mathbf{u}, \tau) + \boldsymbol{\lambda}^T(\mathbf{f}(\mathbf{x}, \mathbf{u}, \tau)) \quad (4.32)$$

$$\Phi(\mathbf{x}, t_f) \stackrel{\text{def}}{=} \boldsymbol{\nu}^T \boldsymbol{\psi}(\mathbf{x}(t_f)) + \phi(\mathbf{x}(t_f)) \quad (4.33)$$

Now consider the perturbation in J_a if we perturb:

- the terminal time to: $t_f + \Delta t$
- the control history to: $\bar{\mathbf{u}}(t) + \delta\mathbf{u}(t)$. Assume that this perturbation in the control history results in a perturbation of the state trajectory to: $\bar{\mathbf{x}}(t) + \delta\mathbf{x}(t)$. Furthermore assume that the perturbed state trajectory satisfies the terminal constraint: $\boldsymbol{\psi}(\mathbf{x} + \delta\mathbf{x})|_{t_f + \Delta t} = \mathbf{0}$ and that the initial condition $\mathbf{x}(t_0)$ remains unchanged.

The perturbation in the augmented cost, after integration by parts of the $\dot{\boldsymbol{\lambda}}$ term, is then given by (see e.g. [8]):

$$\begin{aligned} \Delta J_a = & \left(\frac{\partial \Phi(\mathbf{x}, t_f)}{\partial \mathbf{x}} \dot{\mathbf{x}} + \frac{\partial \Phi(\mathbf{x}, t_f)}{\partial t_f} + \mathcal{L}(\mathbf{x}, \mathbf{u}, t_f) \right) \Bigg|_{\substack{t_f \\ \mathbf{x}(t_f)}} \Delta t \\ & + \left(\frac{\partial \Phi(\mathbf{x}, t_f)}{\partial \mathbf{x}} - \boldsymbol{\lambda}^T \right) \Bigg|_{\substack{t_f \\ \mathbf{x}(t_f)}} \delta\mathbf{x} \\ & \int_{t_0}^{t_f} \left[\left(\frac{\partial \mathcal{H}}{\partial \mathbf{x}} + \dot{\boldsymbol{\lambda}}^T \right) \delta\mathbf{x} + \left(\frac{\partial \mathcal{H}}{\partial \mathbf{u}} \right) \delta\mathbf{u} \right] \mathrm{d}\tau \end{aligned} \quad (4.34)$$

By appropriate choice of the Lagrange multiplier $\boldsymbol{\lambda}$, and the necessary conditions for

$\bar{\mathbf{u}}$ to be optimal, the only remaining terms are:

$$\left(\frac{\partial \Phi(\mathbf{x}, t_f)}{\partial \mathbf{x}}\dot{\mathbf{x}} + \frac{\partial \Phi(\mathbf{x}, t_f)}{\partial t_f} + \mathcal{L}(\mathbf{x}, \mathbf{u}, t_f)\right)\Big|_{\substack{t_f \\ \mathbf{x}(t_f)}} \Delta t \tag{4.35}$$

So that:

$$\begin{aligned} \frac{\partial V(\mathbf{x}, t_0, t_f)}{\partial t_f} &= \lim_{\Delta t \to 0} \frac{\Delta J_a}{\Delta t} && (4.36) \\ &= \left(\frac{\partial \Phi(\mathbf{x}, t_f)}{\partial \mathbf{x}}\dot{\mathbf{x}} + \frac{\partial \Phi(\mathbf{x}, t_f)}{\partial t_f} + \mathcal{L}(\mathbf{x}, \mathbf{u}, t_f)\right)\Big|_{\substack{t_f \\ \mathbf{x}(t_f)}} && (4.37) \end{aligned}$$

Q.E.D.

Remarks:

- Given assumption 4.1.1, $\frac{\partial V}{\partial t_f}$ will be bounded.
- We expect that when $t_f - t_0$ becomes large, the change in cost ΔJ_a and thus $\frac{\partial V}{\partial t_f}$ will become relatively small. In equation (4.28) this should occur through ν changing its value.
- We can recover Michalska and Mayne's result by noting that their result is derived for the case where:
  - The system is time invariant with a single equilibrium point at the origin, viz:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{u}) && (4.38) \\ \mathbf{0} &\equiv \mathbf{f}(\mathbf{0}, \mathbf{0}) && (4.39) \end{aligned}$$

  - There is no explicit terminal penalty, i.e.: $\varphi(\mathbf{x}(t_f) \equiv 0$
  - The terminal constraint is:

$$\psi(\mathbf{x}, t_f) = \mathbf{x}(t_f) = \mathbf{0} \tag{4.40}$$

- The integrand of the cost function is:

$$\mathcal{L}(\mathbf{x}, \mathbf{u}) = \mathbf{x}^T Q \mathbf{x} + \mathbf{u}^T R \mathbf{u} \tag{4.41}$$

where Q, R are both positive definite.

Under these conditions we see that:

- $V(\mathbf{x}, t, t+T) = V(\mathbf{x})$ i.e. the actual cost to drive the system to the origin starting at state $\mathbf{x}$ at time t, depends only on $\mathbf{x}$.
- $V(\mathbf{x})$ is a radially unbounded positive definite function and can serve as a candidate Lyapunov function.
- Because of the terminal constraint $\mathbf{x}(t_f) \equiv 0$, and form of the cost function we have for case where $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + G(\mathbf{x})\mathbf{u}$ that $\mathbf{u}(t_f) = 0$ and $\dot{\mathbf{x}}(t_f) = 0$ and thus:

$$\frac{\partial V(\mathbf{x}, t_0, t_f)}{\partial t_f} = \left(\frac{\partial \Phi(\mathbf{x}, t_f)}{\partial \mathbf{x}} \dot{\mathbf{x}} + \frac{\partial \Phi(\mathbf{x}, t_f)}{\partial t_f} + \mathcal{L}(\mathbf{x}, \mathbf{u}, t_f) \right) \Bigg|_{\substack{t_f \\ \dot{\mathbf{x}}(t_f)}} \tag{4.42}$$

$$= 0 \tag{4.43}$$

- Stability then follows by noting that under these conditions:

$$\frac{\mathrm{d}}{\mathrm{d}t} V(\mathbf{x}, t, t+T) = -(\mathbf{x}^T Q \mathbf{x} + \mathbf{u}^T R \mathbf{u}) \tag{4.44}$$

$$< 0 \quad \forall \mathbf{x} \neq \mathbf{0} \tag{4.45}$$

We can now proceed with the stability result for the nonlinear receding horizon tracking problem. Our goal will be to show that if we apply the receding horizon control law, then "on average" the full state will remain bounded. We take this approach because in general it is not possible to cause the system (4.1) to perfectly track the reference trajectory and at the same time ensure that all the system state variables remain bounded (see e.g. [59]). This means that it generally will not be possible to show that the error variables, say $\mathbf{e} = \mathbf{y} - \mathbf{y}_d$, *and* the "rest of the state", say $\mathbf{x}_r$, tend

to $\mathbf{0}$ as $t \to \infty$.

Lemma 4.3.2 *Assume that we have a system of the form (4.1), and that we have chosen a cost function appropriate for a tracking problem, say:*

$$J = \int_t^{t+T} \mathcal{L}(\mathbf{x}, \mathbf{u}, t) \mathrm{d}\tau \tag{4.46}$$

$$= \int_t^{t+T} (\mathbf{e}^T Q_y \mathbf{e} + \mathbf{x}_r Q_x \mathbf{x}_r + \mathbf{u}^T R \mathbf{u}) \mathrm{d}\tau \tag{4.47}$$

where $\mathbf{e}(\tau) = \mathbf{y}(\tau) - \mathbf{y}_d(\tau)$, *and* Q_y, Q_x, R *are all positive definite and we have a terminal constraint such that* $(\mathbf{e}^T Q_y \mathbf{e} + \mathbf{x}_r Q_x \mathbf{x}_r)_{t_f} = 0$. *Then if Assumption 4.1.1 holds we will have:*

$$\lim_{\eta \to \infty} \frac{1}{\eta} \int_0^{\eta} (\mathbf{e}^T Q_y \mathbf{e} + \mathbf{x}_r Q_x \mathbf{x}_r + \mathbf{u}^T R \mathbf{u}) \mathrm{d}\tau \leq \bar{\kappa} \tag{4.48}$$

for some $\bar{\kappa}$

Proof:

Let $V(\mathbf{x}, t, t+T)$ be the cost incurred if we apply the optimal control over the interval $[t, t+T]$ starting at the state $\mathbf{x}$. Then if we apply the receding horizon control strategy, we know from Lemma 4.3.1 that:

$$\frac{\mathrm{d}}{\mathrm{d}t} V(\mathbf{x}, t, t+T) = -\mathcal{L}(\mathbf{x}, \bar{\mathbf{u}}, t) + \left. \frac{\partial V(\mathbf{x}, t_0, t_f)}{\partial t_f} \right|_{\substack{t_0 = t \\ t_f = t+T}} \tag{4.49}$$

Since assumption 4.1.1 holds we know that $\frac{\partial V(\mathbf{x}, t_0, t_f)}{\partial t_f} \leq \kappa$ for some $\kappa > 0$, so that equation (4.49) becomes:

$$\frac{\mathrm{d}}{\mathrm{d}t} V(\mathbf{x}, t, t+T) \leq -\mathcal{L}(\mathbf{x}, \bar{\mathbf{u}}, t) + \kappa \tag{4.50}$$

Integrating both sides of (4.50) gives:

$$\int_0^{\eta} \frac{\mathrm{d}V}{\mathrm{d}\tau} \mathrm{d}\tau \leq -\int_0^{\eta} \mathcal{L}(\mathbf{x}, \mathbf{u}, \tau) \mathrm{d}\tau + \int_0^{\eta} \kappa \mathrm{d}\tau \tag{4.51}$$

$$\Rightarrow V(\mathbf{x}(\eta),\eta,\eta+T)-V(\mathbf{x}(0),0,T) \leq -\int_0^\eta(\mathbf{e}^TQ_y\mathbf{e}+\mathbf{x}_r^TQ_x\mathbf{x}_r+\mathbf{u}^TR\mathbf{u})\mathrm{d}\tau + \int_0^\eta \kappa \mathrm{d}\tau \tag{4.52}$$

Now by our choice of $\mathcal{L}$ we know that $V(\mathbf{x},t,t+T)\geq 0\ \forall \mathbf{x},t$. Thus:

$$-V(\mathbf{x}(0),0,T) \leq -\int_0^\eta(\mathbf{e}^TQ_y\mathbf{e}+\mathbf{x}_r^TQ_x\mathbf{x}_r+\mathbf{u}^TR\mathbf{u})\mathrm{d}\tau + \int_0^\eta \kappa \mathrm{d}\tau \tag{4.53}$$

$$\Rightarrow \int_0^\eta(\mathbf{e}^TQ_y\mathbf{e}+\mathbf{x}_r^TQ_x\mathbf{x}_r+\mathbf{u}^TR\mathbf{u})\mathrm{d}\tau \leq V(\mathbf{x}(0),0,T)+\int_0^\eta \kappa \mathrm{d}\tau \tag{4.54}$$

$$\Rightarrow \lim_{\eta\to\infty}\frac{1}{\eta}\int_0^\eta(\mathbf{e}^TQ_y\mathbf{e}+\mathbf{x}_r^TQ_x\mathbf{x}_r+\mathbf{u}^TR\mathbf{u})\mathrm{d}\tau \leq \lim_{\eta\to\infty}\left(\frac{V(\mathbf{x}(0),0,T)}{\eta} + \frac{1}{\eta}\int_0^\eta \kappa \mathrm{d}\tau\right) \tag{4.55}$$

$$\Rightarrow \lim_{\eta\to\infty}\frac{1}{\eta}\int_0^\eta(\mathbf{e}^TQ_y\mathbf{e}+\mathbf{x}_r^TQ_x\mathbf{x}_r+\mathbf{u}^TR\mathbf{u})\mathrm{d}\tau \leq \bar{\kappa} \tag{4.56}$$

Q.E.D.

Lemma 4.3.2 provides us with an "integral boundedness" result. We see that "on average" $\mathbf{e}^TQ_y\mathbf{e}+\mathbf{x}_r^TQ_x\mathbf{x}_r$ has to be smaller than κ. Since $\mathbf{y}_d$ is assumed to be bounded, this then implies that $\mathbf{x}$ will "on average" have to be bounded.

4.4 Long Term Optimal Control

The receding horizon controllers discussed in the previous section provide continuous control laws but require much computation: they require that one repeatedly solve an optimal control problem. We have found that the more traditional long term optimal controllers, as defined by equations (4.1), (4.2), (4.3) generally are more useful. These long term controllers are typically used to transfer a system from some initial state to a new equilibrium point. Such will be the case if we want a flexible

robotic manipulator to perform a large scale maneuver such as moving a payload from one location to another. Note that the long-term optimal controller requires one to solve an optimal control problem only once for each maneuver. The receding-horizon control law on the other hand, would in principle require an infinite number of solutions to the optimal control problem for the single maneuver.

The stability of long-term optimal control laws are guaranteed by the following result [2],[45] which also shows that the optimal control laws are robust, even for the case where we are dealing with nonlinear systems.

Theorem 4.4.1 *Consider the system:*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u} \tag{4.57}$$

and take as a cost function:

$$J = \int_{t_0}^{\infty} (m(\mathbf{x}) + \mathbf{u}^T R\mathbf{u}) \mathrm{d}\tau \tag{4.58}$$

where it is assumed that $R > 0$ and $m(\mathbf{x})$ is a radially unbounded globally positive definite function. Assume that:

- *The system (4.57) is completely controllable.*
- *The functions $\mathbf{f}(\mathbf{x}), G(\mathbf{x}), m(\mathbf{x})$ are sufficiently well behaved such that an optimal control exists and that the optimal control satisfies the HJB-equation.*
- *The free system:*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{4.59}$$

$$\mathbf{y} = \mathbf{m}(\mathbf{x}) \tag{4.60}$$

is completely observable.

Then if we apply the optimal control input, say $\bar{\mathbf{u}}$*, the closed loop system will be globally asymptotically stable. Furthermore if we apply the (incorrect) control input:*

$$\mathbf{u} = \bar{\mathbf{u}} + k(t)\bar{\mathbf{u}} \tag{4.61}$$

the closed loop system will still be asymptotically stable provided that:

$$\frac{1}{2} < k(t) < \infty \tag{4.62}$$

4.5 Look-Ahead Control Using the Quasi-Linear Form

4.5.1 Introduction

The control methods discussed in Sections 4.2, 4.3 and 4.4 all require us to solve optimal control problems for a nonlinear system. In this section we construct (and use) approximate solutions to the nonlinear optimization problems by exploiting the quasilinear form for the dynamics. We do this by performing the following steps:

1. Convert our system dynamics to quasilinear form, say:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + G(\mathbf{x})\mathbf{u} \tag{4.63}$$

$$= A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{4.64}$$

2. Estimate the trajectory that the system will follow, for example $\mathbf{x}(t) \approx \hat{\mathbf{x}}(t)$.

3. Treat the quasilinear system as if it were a linear time varying system along the estimated trajectory, viz:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{4.65}$$

$$\approx A(\hat{\mathbf{x}}(t))\mathbf{x} + B(\hat{\mathbf{x}}(t))\mathbf{u} \tag{4.66}$$

$$= \hat{A}(t)\mathbf{x} + \hat{B}(t)\mathbf{u} \tag{4.67}$$

4. Solve the (much easier) optimal control problem for the linear time varying system.

5. Apply the feedback law found from the linear time varying problem to the nonlinear system.

Comparison with standard linearization

Note that the approach outlined above is different from the usual method of linearizing a system along a reference trajectory, and then finding a controller for the linearized system. To see this recall that the process of linearizing a system around a reference trajectory involves the following steps:

- Assume that we have a nonlinear system:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u} \tag{4.68}$$

- Let $\mathbf{u}_n$ be a nominal input which causes a nominal trajectory $\mathbf{x}_n$, i.e.:

$$\dot{\mathbf{x}}_n = \mathbf{f}(\mathbf{x}_n) + G(\mathbf{x}_n)\mathbf{u}_n \tag{4.69}$$

- Now consider applying an input $\mathbf{u}_n + \delta\mathbf{u}$ to (4.68) and write the resulting trajectory as $\mathbf{x}_n + \delta\mathbf{x}$. Equation (4.68) becomes:

$$\dot{\mathbf{x}}_n + \delta\mathbf{x} = \mathbf{f}(\mathbf{x}_n + \delta\mathbf{x}) + G(\mathbf{x}_n + \delta\mathbf{x})(\mathbf{u} + \delta\mathbf{u}) \tag{4.70}$$

- If $\delta\mathbf{x}$ remains small, we can use a Taylor series expansion for the nonlinear functions. If we retain only first order terms we get:

$$\begin{aligned}\dot{\mathbf{x}}_n + \dot{\delta\mathbf{x}} &= \mathbf{f}(\mathbf{x}_n) + \left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right]_{\mathbf{x}_n} \delta\mathbf{x} \\ &\quad + G(\mathbf{x}_n)\mathbf{u}_n + \left[\frac{\partial}{\partial \mathbf{x}}(G(\mathbf{x})\mathbf{u}_n)\right]_{\mathbf{x}_n} \delta\mathbf{x} + G(\mathbf{x}_n)\delta\mathbf{u} \qquad (4.71) \\ \Rightarrow \dot{\delta\mathbf{x}} &= \left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}} + \frac{\partial}{\partial \mathbf{x}}(G(\mathbf{x})\mathbf{u}_n)\right]_{\mathbf{x}_n} \delta\mathbf{x} + G(\mathbf{x}_n)\delta\mathbf{u} \qquad (4.72)\end{aligned}$$

- The linear time varying system that we obtain through linearization is given by:

$$\dot{\delta\mathbf{x}} = A(t)\delta\mathbf{x} + B(t)\delta\mathbf{u} \qquad (4.73)$$

where:

$$A(t) \stackrel{\text{def}}{=} \left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}} + \frac{\partial}{\partial \mathbf{x}}(G(\mathbf{x})\mathbf{u}_n)\right]_{\mathbf{x}_n} \qquad (4.74)$$

$$B(t) \stackrel{\text{def}}{=} G(\mathbf{x}_n) \qquad (4.75)$$

- We now find a control law for the linearized system, say:

$$\delta\mathbf{u} = K(t)\delta\mathbf{x} \qquad (4.76)$$

- To control the full nonlinear system we then add the perturbation control input $\delta\mathbf{u}$, to the nominal control, viz:

$$\mathbf{u} = \mathbf{u}_n + K(t)\delta\mathbf{x} \qquad (4.77)$$

The essential difference between our approach and the standard linearization approach is that the latter requires $\mathbf{u}_n$ to be known a priori, while the approach based on the quasilinear form will generate the full $\mathbf{u}$ in one step.

4.5.2 The Look-ahead Issue

As indicated earlier the main difference between the methods of this chapter and the previous chapter is that we now use knowledge about the future behavior of $A(\mathbf{x})$ and $B(\mathbf{x})$ in the control laws. To clarify the need for look-ahead let us examine the linear time varying quadratic regulator problem where the system dynamics are:

$$\dot{\mathbf{x}} = A(t)\mathbf{x} + B(t)\mathbf{u} \tag{4.78}$$

and the cost function is:

$$J = \mathbf{x}(t_f)^T H \mathbf{x}(t_f) + \int_{t_0}^{t_f} (\mathbf{x}^T Q \mathbf{x} + \mathbf{u}^T R \mathbf{u}) \mathrm{d}\tau \tag{4.79}$$

It can be shown [28] that the optimal control input is given by:

$$\mathbf{u}(t) = -R^{-1} B(t)^T P(t, t_f)\mathbf{x} \tag{4.80}$$

where:

$$\begin{aligned} -\frac{\partial P(\tau, t_f)}{\partial \tau} &= Q + P(\tau, t_f)A(\tau) + A^T(\tau)P(\tau, t_f) \\ &\quad -P(\tau, t_f)B(\tau)R^{-1}B^T(\tau)P(\tau, t_f) \qquad (4.81) \\ P(t_f, t_f) &= H \qquad (4.82) \end{aligned}$$

I.e. to obtain the optimal control input, we have to calculate $P(t, t_f)$ by integrating the differential equation (4.81) *backwards* in time from the final time t_f, to the current time t. This clearly shows that the control at the current time t depends on the future behavior of the matrices $A(t), B(t)$. (Note that this dependence of the optimal control input on the future behavior of the system is also reflected for the more general case by the fact that the HJB-equation has a boundary condition imposed at the terminal time, see e.g. Section (3.2.1)).

Having established the need for look-ahead we have to address a further issue:

The fundamental difference between linear time varying systems:

$$\dot{\mathbf{x}} = A(t)\mathbf{x} + B(t)\mathbf{u} \tag{4.83}$$

and quasilinear systems

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{4.84}$$

is that for linear systems the time histories $A(t), B(t)$ are always the same no matter what initial condition we use. On the other hand for the quasilinear systems the time histories for $A(\mathbf{x}(t)), B(\mathbf{x}(t))$ will generally be different for each initial condition — we can think of the quasilinear system as an ensemble of linear time varying systems.

It is this "unpredictability" of $A(\mathbf{x}), B(\mathbf{x})$ that in general makes it more difficult to obtain controllers for quasilinear systems. To overcome this problem of not knowing the future behavior of $A(\mathbf{x}), B(\mathbf{x})$ we will:

- Define an optimal control problem which will cause the system state to follow as closely as possible a specified trajectory, say $\mathbf{x}_d$.
- Assume that this desired trajectory is achievable and use it to estimate the behavior of $A(\mathbf{x}), B(\mathbf{x})$,viz:

$$A(\mathbf{x}) \approx A(\mathbf{x}_d) \tag{4.85}$$

$$B(\mathbf{x}) \approx B(\mathbf{x}_d) \tag{4.86}$$

In essence it is by defining/solving an appropriate optimization problem that we are able to estimate the future behavior of the system and thus use look-ahead.

4.5.3 Implementing the Look-ahead Controllers

In this section we further detail the method used to implement the look-ahead control laws. We will focus the method we have found most useful, i.e. to (approximately)

implement a long term optimal controller (see Section 4.4) which transfers the system from an initial state to an equilibrium point. A brief discussion on how we can use a similar approach for the short-term tracker and receding horizon tracker problems will conclude the section. In order to more easily discuss the methodology we will again assume (although it is not required in general) that the state vector explicitly contains the output variables, viz:

$$\mathbf{x} = \begin{bmatrix} \mathbf{y} \\ \mathbf{x}_r \end{bmatrix} \tag{4.87}$$

The look-ahead control laws we use, have a similar objective to the controllers discussed in Section 3.2.3, i.e. we want the plant output variables, say $\mathbf{y}$, to track desired values, say $\mathbf{y}_d$, and at the same time ensure that the rest of the state variables $\mathbf{x}_r$ remain small. With this goal in mind we construct a quadratic optimization problem similar to that discussed in Section 3.2.3, viz:

Assume that the desired trajectory is generated by an autonomous system (typically linear time invariant):

$$\dot{\mathbf{x}}_d = A_d \mathbf{x}_d \tag{4.88}$$

$$\mathbf{y}_d = C_d \mathbf{x}_d \tag{4.89}$$

Form the augmented system dynamics:

$$\dot{\mathbf{z}} = F(\mathbf{z})\mathbf{z} + G(\mathbf{z})\mathbf{u} \tag{4.90}$$

where:

$$\mathbf{z} = \begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{x}}_d \end{bmatrix} \quad F = \begin{bmatrix} A(\mathbf{x}) & 0 \\ 0 & A_d \end{bmatrix} \quad G = \begin{bmatrix} B(\mathbf{x}) \\ 0 \end{bmatrix} \mathbf{u} \tag{4.91}$$

Find the control input that minimizes the cost function:

$$J = \mathbf{z}^T(t_f)H\mathbf{z}(t_f) + \int_{t_0}^{t_f}((\mathbf{y}-\mathbf{y}_d)^T Q_y(\mathbf{y}-\mathbf{y}_d) + \mathbf{x}_r^T Q_x \mathbf{x}_r + \rho \mathbf{u}^T R\mathbf{u})\mathrm{d}\tau \quad (4.92)$$

$$\stackrel{\text{def}}{=} \mathbf{z}^T(t_f)H\mathbf{z}(t_f) + \int_{t_0}^{t_f}(\mathbf{z}^T Q\mathbf{z} + \rho \mathbf{u}^T R\mathbf{u})\mathrm{d}\tau \quad (4.93)$$

The look-ahead control law is then implemented by finding an approximate solution to this problem. We first estimate $\mathbf{z}(t)$ by say $\hat{\mathbf{z}}(t)$, for the interval $[t_0, t_f]$, and then integrate the Ricatti ODE

$$-\frac{\partial P(\tau,t_f)}{\partial \tau} = Q + P(\tau,t_f)F(\hat{\mathbf{z}}(\tau)) + F^T(\hat{\mathbf{z}}(\tau))P(\tau,t_f) - \frac{1}{\rho}P(\tau,t_f)G(\hat{\mathbf{z}}(\tau))R^{-1}G^T(\hat{\mathbf{z}}(\tau))P(\tau,t_f) \quad (4.94)$$

backwards in time starting from the terminal condition:

$$P(t_f,t_f) = H \quad (4.95)$$

and then apply the (approximately optimal) control input:

$$\mathbf{u} = -\frac{1}{\rho}R^{-1}G^T(\hat{\mathbf{z}})P(t,t_f)\mathbf{z} \quad (4.96)$$

to the full nonlinear system (4.90).

Remarks:

- By using a cost function of the form (4.93) and letting $\rho \to 0$ the system will follow the desired trajectory and at the same time keep $\mathbf{x}_r$ small, if this is at all possible. Thus when ρ is small we will generally be able to better predict the actual trajectory.
- Since $\mathbf{y}_d$ in equation (4.89) will be a sum of complex exponentials we can generate a very wide variety of desired trajectories by appropriately changing the dynamics in equation (4.88).

- If our goal is to transfer the system from an initial condition to an equilibrium state, we have found it expedient to do the following. Use $t_f \approx 5T$ where T is the largest time constant of the trajectory generator (4.88), and use $H = \bar{P}$ where $\bar{P}$ is the positive semi-definite solution to the steady state Ricatti equation:

$$0 = Q + \bar{P}F(\mathbf{0}) + F^T(\mathbf{0})\bar{P} - \frac{1}{\rho}\bar{P}G(\mathbf{0})R^{-1}G^T(\mathbf{0})\bar{P} \tag{4.97}$$

This can be explained as follows. Transferring the system to a new equilibrium condition requires that $\mathbf{y}_d \rightarrow \mathbf{0}$ as $t \rightarrow \infty$. If the controller is capable of keeping $\mathbf{x}_r$ relatively small, we expect that at $t \approx 5T$ we will have the full state $\mathbf{z} \approx \mathbf{0}$. Once we are at $\mathbf{z} \approx \mathbf{0}$ we can use a stable controller based on the linear approximation of the plant dynamics. As discussed in Chapter 3, this approximation near the origin is given by:

$$\dot{\mathbf{z}} = F(\mathbf{0})\mathbf{z} + G(\mathbf{0})\mathbf{u} \tag{4.98}$$

By using:

$$\mathbf{u} = -\frac{1}{\rho}R^{-1}G(\mathbf{0})^T\bar{P}\mathbf{z} \tag{4.99}$$

for all $t \geq 5T$ we are then simply using the Linear Quadratic Regulator feedback law based on the linearized dynamics. Note that the choice of $H = \bar{P}$ ensures that the nonlinear control that was used for $t \leq 5T$ tends smoothly toward the linear control law used for $t \geq 5T$.

- One method to estimate the behavior of $\mathbf{z}(t)$ is to set:

$$\hat{\mathbf{z}} = \begin{bmatrix} \mathbf{y}_d \\ \mathbf{0} \\ \mathbf{y}_d \end{bmatrix} \tag{4.100}$$

since we expect $\mathbf{x}_r \approx \mathbf{0}$ and $\mathbf{y} \approx \mathbf{y}_d$. However we have found that for the long term optimal controllers the following method provided better results:

- At the initial condition, say $\mathbf{z}_0$, solve the steady state Ricatti equation:

$$0 = Q + P_0 F(\mathbf{z}_0) + F(\mathbf{z}_0)^T P_0 - \frac{1}{\rho} P_0 G(\mathbf{z}_0) R^{-1} G^T(\mathbf{z}_0) P_0 \qquad (4.101)$$

- Estimate the future trajectory of $\mathbf{z}$ by calculating the response of the linear time invariant system:

$$\dot{\hat{\mathbf{z}}} = \left(F(\mathbf{z}_0) - \frac{1}{\rho} G(\mathbf{z}_0) R^{-1} G^T(\mathbf{z}_0) P_0 \right) \hat{\mathbf{z}} \qquad (4.102)$$

(Note that this method of estimating $\mathbf{z}$ is consistent with the theorem by Kokotovic et al discussed in Section 3.3.3).

Look-Ahead Controllers for the Short-Term Tracking and Receding Horizon Problems

For both the short-term tracking problem (Section 4.2) and the receding horizon control laws (Section 4.3) we have to (repeatedly) solve an optimal control problem subject to a terminal constraint. For these cases we have found a penalty function approach [15] to be useful. The penalty function method typically uses a cost function of the form:

$$J = \mu_f \, \boldsymbol{\psi}(\mathbf{x})^T \boldsymbol{\psi}(\mathbf{x})\Big|_{t_f} + \int_{t_0}^{t_f} \mathcal{L}(\mathbf{x}, \mathbf{u}) \mathrm{d}\tau \qquad (4.103)$$

and ensures that the terminal constraint $\boldsymbol{\psi}(\mathbf{x}) = \mathbf{0}$ is (approximately) met by making μ_f large. Given this form of the cost function we can then compute the approximate optimal control law in the same manner that we used for the long-term look-ahead controller as discussed above.

4.6 Simulation Results

In this section we will apply the look-ahead controllers to the one-link arm example that was discussed in Chapter 3. We see that by using approximate look-ahead, all three methods, i.e.:

- short-term optimal control (Section 4.2)
- receding horizon control (Section 4.3)
- and the long-term look-ahead controller (Section 4.4)

result in stable responses for the case where the continuous Ricatti Design was unstable.

In the following examples we will use the system dynamics as in Section 3.6, where we had:

$$\dot{\mathbf{z}} = F(\mathbf{z})\mathbf{z} + G(\mathbf{z})u \tag{4.104}$$

with state vector:

$$\mathbf{z} = \begin{bmatrix} \theta & q & \dot{\theta} & \dot{q} & \mathbf{x}_d^T \end{bmatrix} \tag{4.105}$$

and dynamics matrices:

$$F(\mathbf{z}) = \begin{bmatrix} A(\mathbf{x}) & 0 \\ 0 & A_d \end{bmatrix} \quad G(\mathbf{z}) = \begin{bmatrix} B \\ 0 \end{bmatrix} \tag{4.106}$$

where:

$$A(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\alpha_{11} + \alpha_{12}\dot{\theta}^2 & 0 & 0 \\ 0 & -\alpha_{21} + \alpha_{22}\dot{\theta}^2 & 0 & 0 \end{bmatrix} \tag{4.107}$$

$$B(\mathbf{x}) = \begin{bmatrix} 0 \\ 0 \\ -\beta_1 \\ -\beta_2 \end{bmatrix} \tag{4.108}$$

$$A_d = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} \tag{4.109}$$

We used the values of $\alpha_{ij}, \beta_i, \omega_n$ and ζ which resulted in an unstable continuous Ricatti design controller, viz:

$$\alpha_{11} = 20 \tag{4.110}$$

$$\alpha_{12} = 0.12 \tag{4.111}$$

$$\alpha_{21} = -26 \tag{4.112}$$

$$\alpha_{22} = 170 \tag{4.113}$$

$$\beta_1 = 8 \times 10^{-5} \tag{4.114}$$

$$\beta_2 = 1.4 \times 10^{-3} \tag{4.115}$$

$$\omega_n = 10 \tag{4.116}$$

$$\zeta = 1 \tag{4.117}$$

Example 4.6.1 [Short Term Optimal Tracker]

Figure 4-2 shows the simulation results when we used the look-ahead method to find an approximately optimal short term tracking control input. We used a similar cost function to Section 3.6, Case 3, except that now we added a terminal penalty so that the terminal constraint would (approximately) be met. The cost function for this example was:

$$J = \mu_f(\mathbf{e}^T Q_{yf}\mathbf{e} + \mathbf{z}^T Q_{zf}\mathbf{z}) + \int_{t_i}^{t_i+T} (\mathbf{e}^T Q_y \mathbf{e} + \mathbf{z}^T Q_z \mathbf{z} + \rho^2 u^2) \mathrm{d}\tau \tag{4.118}$$

where:

$$T = 0.2 \tag{4.119}$$

$$\mu = 100 \tag{4.120}$$

$$Q_{yf} = \text{diag}\begin{bmatrix} 1 \times 10^7 & 1 \times 10^5 \end{bmatrix} \tag{4.121}$$

$$Q_{zf} = \text{diag}\begin{bmatrix} 0 & 1 \times 10^5 & 0 & 1 \times 10^3 & 0 & 0 \end{bmatrix} \tag{4.122}$$

$$Q_y = \text{diag}\begin{bmatrix} 1 \times 10^7 & 1 \times 10^5 \end{bmatrix} \tag{4.123}$$

$$Q_z = \text{diag}\begin{bmatrix} 0 & 1 \times 10^5 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.124}$$

$$\rho = 1 \times 10^{-4} \tag{4.125}$$

We see from Figure 4-2 that the system's response reflects the discontinuity in the short term tracking control law at the boundaries of the time intervals, i.e. $t = 0.2s, t = 0.4s, t = 0.6s$.

Example 4.6.2 (Receding Horizon Tracker)

Figure 4-3 shows the simulation results when we used the look-ahead method to find an approximation to the receding horizon control law discussed in Section 4.3. We used the same cost function etc. as in the previous example except that the terminal penalty was changed to:

$$\mu = 100 \tag{4.126}$$

$$Q_{yf} = \text{diag}\begin{bmatrix} 1 \times 10^7 & 0 \end{bmatrix} \tag{4.127}$$

$$Q_{zf} = \text{diag}\begin{bmatrix} 0 & 1 \times 10^5 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.128}$$

$$\tag{4.129}$$

For this example we used a look-ahead interval of $T = 0.2s$ which is two times the time constant of the desired dynamics. Note that we obtained a smooth closed loop response from the system. However the computational cost was large. In principle

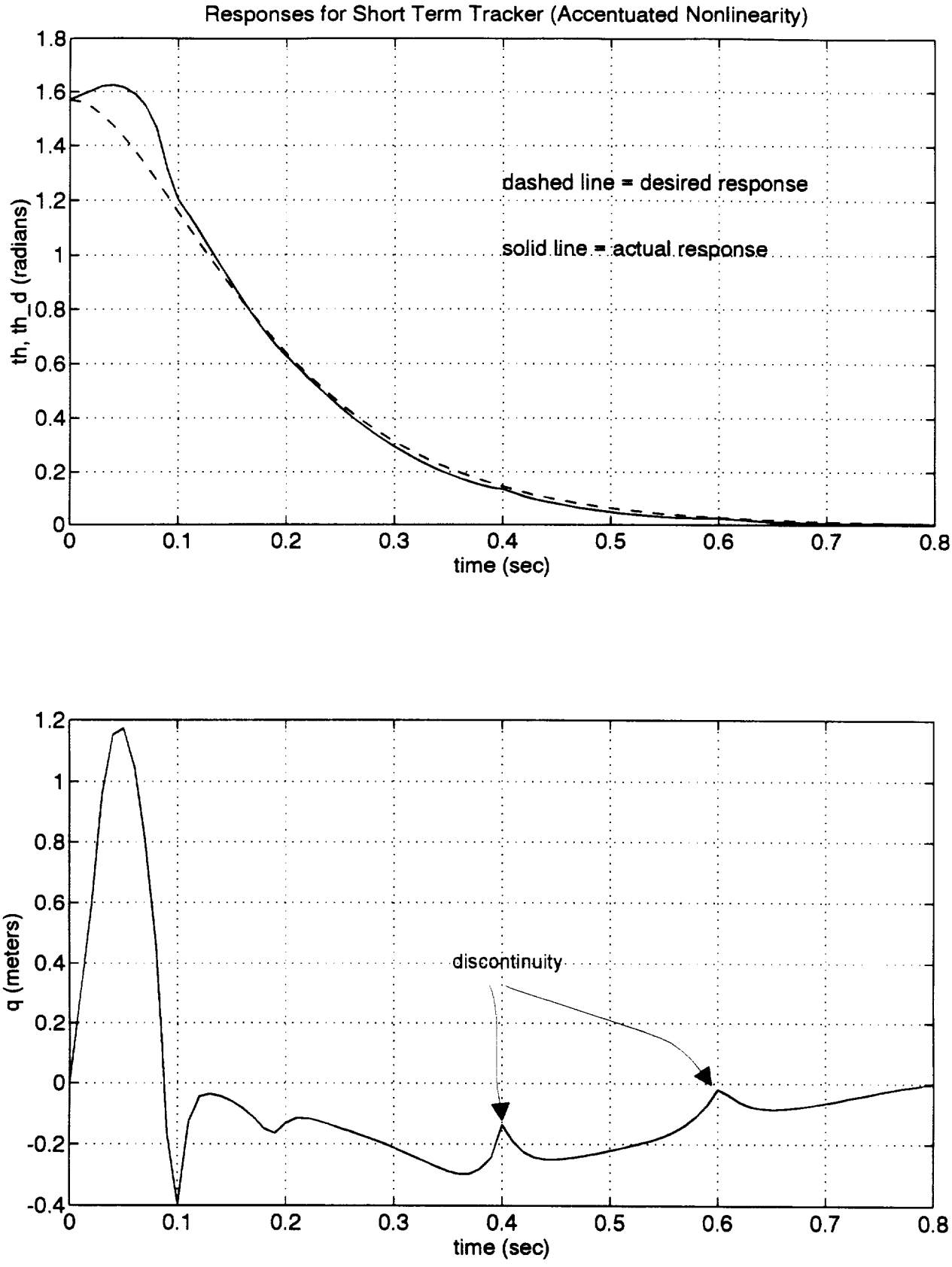


Figure 4-2: Responses for Short Term Tracker

the receding horizon method requires us to repeatedly solve the optimal control problem for each state as we move along the trajectory. In practice we re-compute the optimal control at short intervals along the trajectory as in this example, where we re-computed the optimal control each $0.01s$.

Example 4.6.3 (Long-Term Look-ahead)

Figure 4-4 shows the results when we used the long-term look-ahead controller discussed in Section 4.5.3. We used the same cost function as in Section 3.6, Case 3. To estimate the trajectory we used the method discussed in Section 4.5.3, i.e. we solved the steady state Ricatti equation at the initial state, computed the response for the linear time invariant system of equation 4.102 and used $\hat{\mathbf{z}}(t)$ as our estimate of the system trajectory. We then integrated the Ricatti differential equation (4.94) backwards in time. To start the integration process we used $P(t_f, t_f) = \bar{P}$ where $\bar{P}$ was the solution the steady state Ricatti equation at the terminal state, i.e. $\mathbf{z} = \mathbf{0}$.

4.7 Summary

In this Chapter we examined methods for controlling nonlinear systems that utilized "knowledge" about the future behavior of the system, viz:

- Short term optimal control.
- Receding horizon control.
- Long-term optimal control.

All three controllers gave stable system responses for a case where the zero look-ahead control law discussed in the previous Chapter was unstable. Of the three methods we preferred the long-term optimal control approach, because the short term optimal

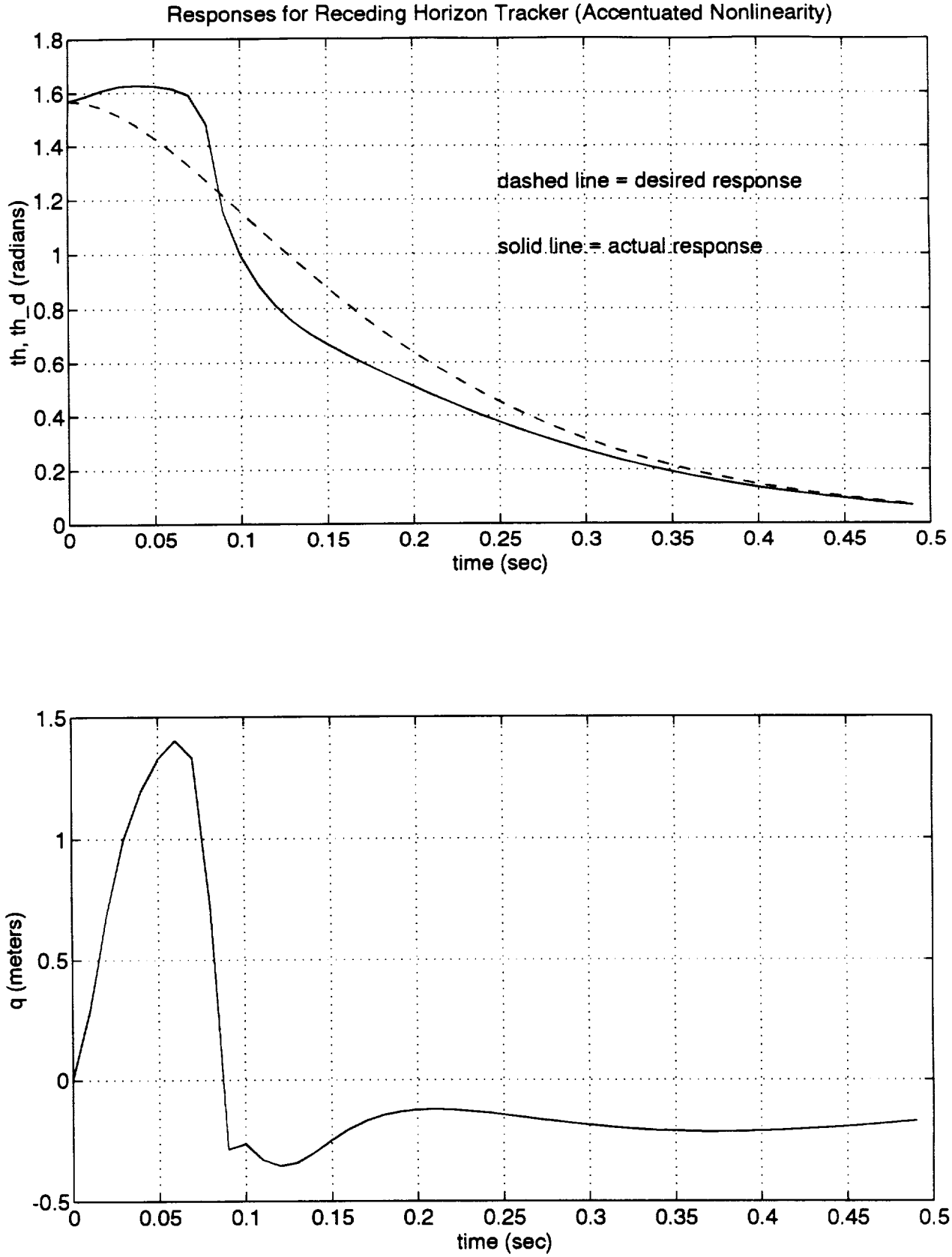
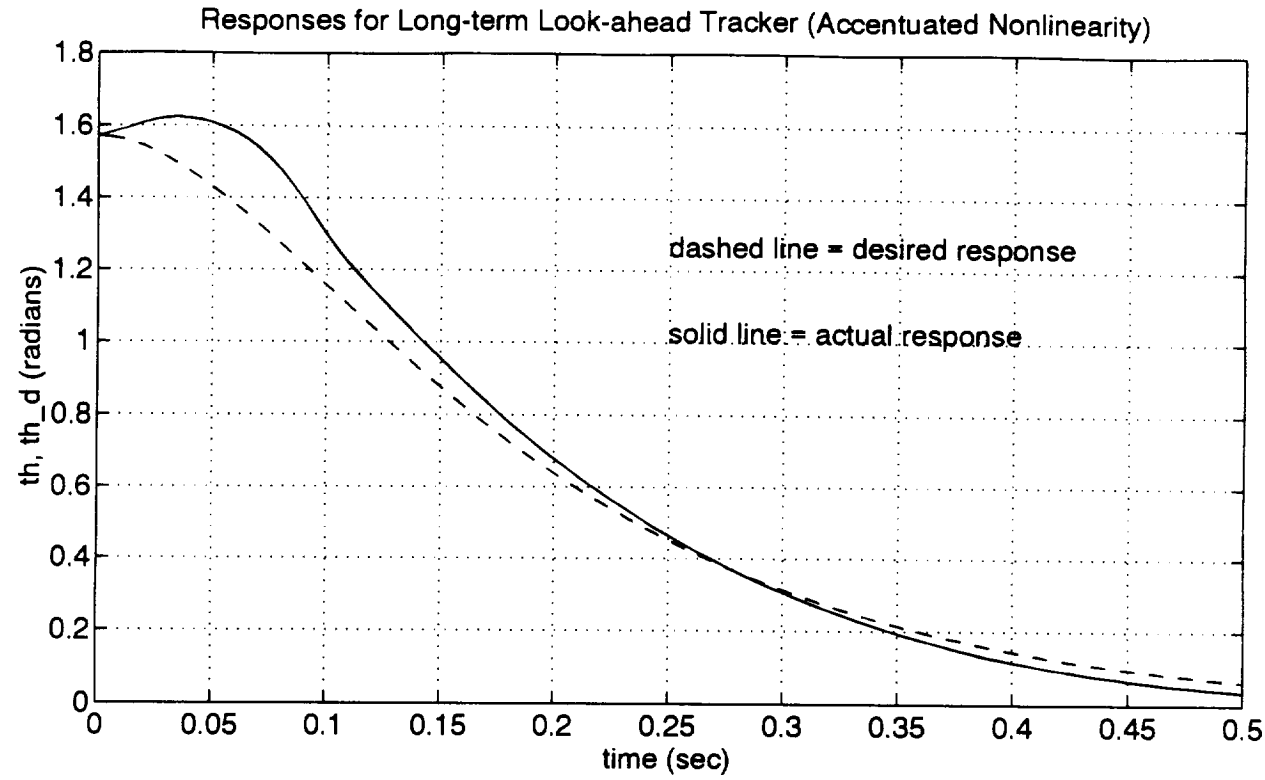


Figure 4-3: Responses for Receding Horizon Tracker



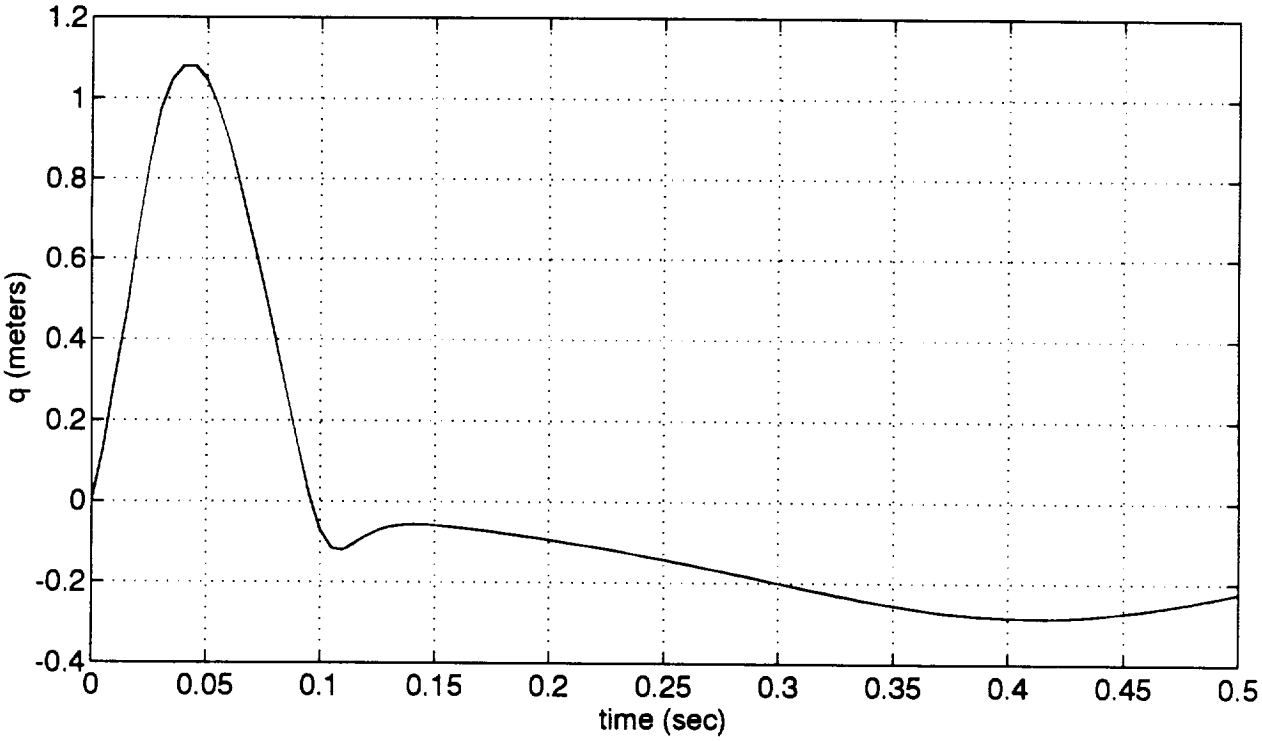


Figure 4-4: Responses for Long Term Look-ahead Controller

controllers resulted in discontinuous control laws, while the receding horizon control laws were undesirable because of the amount of computation required.

Chapter 5

Control of Flexible Manipulators

5.1 Introduction

In this chapter we use the methods discussed in Chapters 3 and 4 to develop controllers for a complex nonlinear dynamical system: a flexible manipulator similar to the space shuttle Remote Manipulator System[20]. Since flexible manipulators are in general not feedback linearizable the standard methods for controlling rigid manipulators cannot be applied (we assume that the system will have more degrees of freedom than actuators). Several researchers (see e.g. [48], [6],[10],[49]) have investigated the problem of controlling flexible manipulators but at present no mature methodology is available that will guarantee global closed loop asymptotic stability. The methods we use to control the system also do not guarantee global closed loop stability, but do provide us with a generic approach to deal with the full nonlinear nature of the system and have resulted in well behaved closed loop systems.

5.2 System Dynamics

To derive the equations of motion for the flexible manipulator we used the method described in Appendix A, i.e. we used Lagrange's equations and modeled the flexibility using an assumed modes method. We made the following assumptions:

- The manipulator has two flexible links.
- The base of the manipulator is stationary in inertial space.
- The gross motion angles, θ_1, θ_2 measure the relative angle between the links (i.e. we use the "relative-co-ordinates" defined in Appendix A). This means that θ_1 measures the angle of the inboard link relative to an inertial reference frame, and θ_2 measures the angle of the outboard link relative to a line tangent to the tip of the inboard link.
- The structural flexibility is modeled using an assumed modes approach. Each link has one flex-mode with a sinusoidal mode shape, i.e. the deflection of a "slice" on a link (see figure 5-1) is given by:

$$v(\zeta, q) = q\varphi(\zeta) \tag{5.1}$$

$$= q \sin\left(\frac{\pi\zeta}{L}\right) \tag{5.2}$$

  A consequence of using these mode shapes is that the gross-motion angles θ_1, θ_2 represent angles measured relative to imaginary lines joining the root and tip of each link. Thus by controlling θ_1 and θ_2 we will be directly controlling the tip motion.

- The arm has a payload of $3000lb$ attached to the tip of the outboard link. The payload is considered to be a point mass, with no rotational inertia.
- The effects of gravity are not included in the model since it is assumed that the manipulator will be operating in space.

- Other physical parameters are [27]:

$$\text{Stiffness } (EI) = 1.378944 \times 10^{11} * 2.081157 \times 10^{-5} \frac{N}{m^2} \quad (5.3)$$

$$\text{Density/Unit Length } (\rho_A) = 55.16 \frac{kg}{m} \quad (5.4)$$

$$\text{Length of Link 1 } (L_1) = 6.37m \quad (5.5)$$

$$\text{Length of Link 2 } (L_2) = 7.1m \quad (5.6)$$

- The control inputs to the system are torque motors located at the root of each link.

Using the assumptions above, the equations of motion for the system were automatically generated using the routines described in Appendix A. The resulting system dynamics were obtained in quasilinear form, viz:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \quad (5.7)$$

where the state vector is:

$$\mathbf{x} = \begin{bmatrix} \theta_1 \\ q_1 \\ \theta_2 \\ q_2 \\ \dot{\theta}_1 \\ \dot{q}_1 \\ \dot{\theta}_2 \\ \dot{q}_2 \end{bmatrix} = \begin{bmatrix} \text{gross motion angle - link 1} \\ \text{flex mode deflection - link 1} \\ \text{gross motion angle - link 2} \\ \text{flex mode deflection - link 2} \\ \text{gross motion angular rate - link 1} \\ \text{flex mode deflection rate - link 1} \\ \text{gross motion angular rate - link 2} \\ \text{flex mode deflection rate - link 2} \end{bmatrix} \quad (5.8)$$

and the system matrices are given by:

$$A(\mathbf{x}) = \begin{bmatrix} 0 & I \\ H^{-1}(\mathbf{q})K & H^{-1}(\mathbf{q})C(\mathbf{q},\dot{\mathbf{q}}) \end{bmatrix} \quad B(\mathbf{x}) = \begin{bmatrix} 0 \\ H^{-1}(\mathbf{q})\Gamma(\mathbf{q}) \end{bmatrix} \quad (5.9)$$

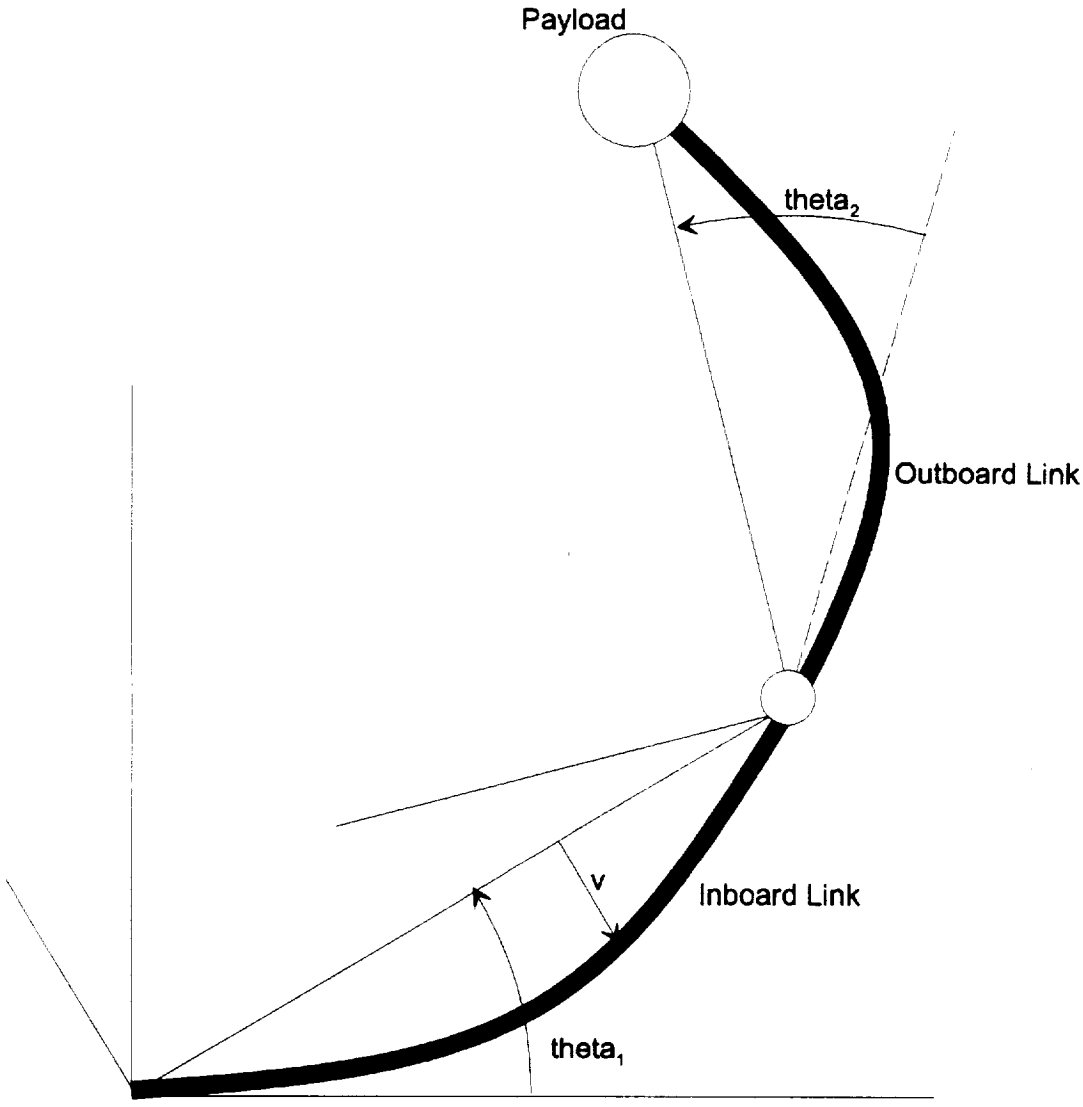


Figure 5-1: Two Link Flexible Manipulator

with:

$$H(\mathbf{q}) = \text{Inertia matrix} \tag{5.10}$$

$$K = \text{Stiffness matrix} \tag{5.11}$$

$$C(\mathbf{q}, \dot{\mathbf{q}}) = \text{"Coriolis" Matrix (see Appendix A)} \tag{5.12}$$

and $\Gamma(\mathbf{q})$ is determined by examining the generalized forces [17] acting on the system. For the parameters given above, the linearized system dynamics has four poles at the origin, and two sets of undamped poles with frequencies $\approx$ 8Hz and $\approx$ 12Hz respectively.

The equations of motion above appear to be quite simple, but in fact represent complex dynamics as can be seen by examining equations (5.13),and (5.14) which give the numerator and denominator of the second-row, third column element of the inverse of the inertia matrix. It is this complexity that makes the routines described in Appendix A useful, since they can produce output in symbolic form suitable for documentation purposes, as well as directly generate "C" or Fortran code that can be, for example, used in simulation models.

$$\begin{aligned}
\text{num}[H^{-1}]_{23} = {} & \left\{18\,\rho_A^2\pi^5 q_{21}^2 - \left(-72\,\pi^3 + 12\,\pi^5\right)\rho_A^2 L_2^2 + 36\,\rho_A\,\pi^5 m_2\,L_2\right\} q_{11}^2 \\
& + \left\{\left(36\,\pi^3 + 12\,\pi^5\right)\rho_A^2 L_1^2 + \left(\left(36\,\pi^5 - 288\,\pi^3\right)\rho_A^2 L_2\right.\right. \\
& \left.\left. + 36\,\rho_A\,\pi^5 m_2\right) L_1\right\} q_{21}^2 - \left\{-144\,\rho_A^2 L_1^3 \pi^2 \sin(\theta_2 - \frac{\pi\, q_{11}}{L_1}) \right. \\
& + \left(\left(36\,\pi^4 \sin(2\,\theta_2 - \frac{2\,\pi\, q_{11}}{L_1}) + 288\,\pi^2 \sin(\frac{2\,\pi\, q_{11}}{L_1} - 2\,\theta_2)\right.\right. \\
& \left.\left. -36\,\pi^4 \sin(\frac{2\,\pi\, q_{11}}{L_1} - 2\,\theta_2) - 288\,\pi^2 \sin(2\,\theta_2 - \frac{2\,\pi\, q_{11}}{L_1})\right)\rho_A^2 L_2^2 \right. \\
& \left. -144\,L_2\,\pi^4 \rho_A\, m_2 \sin(\frac{2\,\pi\, q_{11}}{L_1} - 2\,\theta_2)\right) L_1\Big\}\, q_{21} \\
& - \left\{\left(-288\,\pi\,\cos(\theta_2 - \frac{\pi\, q_{11}}{L_1}) + 36\,\pi^3 \cos(\theta_2 - \frac{\pi\, q_{11}}{L_1})\right)\rho_A^2 L_2\right. \\
& \left. +72\,\rho_A\, m_2\,\pi^3 \cos(-\theta_2 + \frac{\pi\, q_{11}}{L_1})\right\} L_1^3
\end{aligned}$$

$$
\begin{aligned}
&- \left\{ \left(-144\,\pi + 8\,\pi^5 - 24\,\pi^3 \right) \rho_A^2 L_2^2 + \left(24\,\pi^5 m_2 + 72\,\pi^3 m_2 \right) \rho_A\, L_2 \right\} L_1^2 \\
&+ \left\{ \left(48\,\pi^3 \cos(2\,\theta_2 - \frac{2\,\pi\,q_{11}}{L_1}) - 9\,\pi^5 \cos(2\,\theta_2 - \frac{2\,\pi\,q_{11}}{L_1}) \right.\right. \\
&\left. +15\,\pi^5 - 96\,\pi^3 \right) \rho_A^2 L_2^3 + \left(60\,\pi^5 m_2 - 36\,\pi^5 m_2\, \cos(\frac{2\,\pi\,q_{11}}{L_1} - 2\,\theta_2) \right. \\
&+288\,m_2\,\pi^3 \cos(\frac{2\,\pi\,q_{11}}{L_1} - 2\,\theta_2) - 144\,\pi^3 m_2 \\
&\left. -288\,m_2\,\pi^3 \cos(2\,\theta_2 - \frac{2\,\pi\,q_{11}}{L_1}) \right) \rho_A\, L_2^2 \\
&\left. - \left(-36\,m_2^2 \pi^5 \cos(\frac{2\,\pi\,q_{11}}{L_1} - 2\,\theta_2) + 36\,m_2^2 \pi^5 \right) L_2 \right\} L_1 \qquad (5.13)
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{den}[H^{-1}]_{23} \;=\; & \left\{ 9\,\rho_A^3 L_1^2 \pi^4 q_{21}^2 + \left(\left(6\,\pi^4 - 36\,\pi^2 \right) \rho_A^3 L_2^2 + 18\,\rho_A^2 \pi^4 m_2\, L_2 \right) L_1^2 \right\} q_{11}^2 \\
&+ \left\{ \left(6\,\pi^4 - 36\,\pi^2 \right) \rho_A^3 L_1^4 + \left(\left(18\,\pi^4 - 144\,\pi^2 \right) \rho_A^3 L_2 + 18\,\rho_A^2 \pi^4 m_2 \right) L_1^3 \right\} q_{21}^2 \\
&- \left\{ \left(-18\,\pi^3 \sin(\frac{2\,\pi\,q_{11}}{L_1} - 2\,\theta_2) + 18\,\pi^3 \sin(2\,\theta_2 - \frac{2\,\pi\,q_{11}}{L_1}) \right.\right. \\
&\left. -144\,\pi\, \sin(2\,\theta_2 - \frac{2\,\pi\,q_{11}}{L_1}) + 144\,\pi\, \sin(\frac{2\,\pi\,q_{11}}{L_1} - 2\,\theta_2) \right) \rho_A^3 L_2^2 \\
&\left. -72\,\rho_A^2 L_2\,\pi^3 m_2\, \sin(\frac{2\,\pi\,q_{11}}{L_1} - 2\,\theta_2) \right\} L_1^3 q_{21} \\
&+ \left\{ \left(4\,\pi^4 + 144 - 48\,\pi^2 \right) \rho_A^3 L_2^2 - \left(-72\,\pi^2 m_2 + 12\,\pi^4 m_2 \right) \rho_A^2 L_2 \right\} L_1^4 \\
&+ \left\{ \left(\frac{15\,\pi^4}{2} - 48\,\pi^2 + 24\,\pi^2 \cos(2\,\theta_2 - \frac{2\,\pi\,q_{11}}{L_1}) \right.\right. \\
&\left. -9\,\pi^4 \cos(2\,\theta_2 - \frac{2\,\pi\,q_{11}}{L_1}) 1/2 \right) \rho_A^3 L_2^3 \\
&- \left(-144\,m_2\,\pi^2 \cos(2\,\theta_2 - \frac{2\,\pi\,q_{11}}{L_1}) - 18\,\pi^4 m_2\, \cos(\frac{2\,\pi\,q_{11}}{L_1} - 2\,\theta_2) \right. \\
&+30\,\pi^4 m_2\, -72\,\pi^2 m_2 + 144\,m_2\,\pi^2 \cos(\frac{2\,\pi\,q_{11}}{L_1} - 2\,\theta_2) \Big) \rho_A^2 L_2^2 \\
&\left. + \left(18\,m_2^2 \pi^4 - 18\,m_2^2 \pi^4 \cos(\frac{2\,\pi\,q_{11}}{L_1} - 2\,\theta_2) \right) \rho_A\, L_2 \right\} L_1^3 \qquad (5.14)
\end{aligned}
$$

5.3 Simulation Results

To control the flexible manipulator described above, we used the two methods we found most useful, i.e. the continuous Ricatti design method described in Chapter 3 and the long-term look-ahead controller described in Sections 4.4 and 4.5. In both cases we included terms to obtain integral control.

Basic Optimization Problem

The basic structure of the controller is derived from the combined tracking and regulator problem discussed in Section 3.2.3. For this example we used the following state space equations for the augmented system:

$$\dot{\mathbf{z}} = F(\mathbf{z})\mathbf{z} + G(\mathbf{z})\mathbf{u} \tag{5.15}$$

with state vector:

$$\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_d \\ \mathbf{e}_I \end{bmatrix} = \begin{bmatrix} \text{flexible manipulator states, see equation (5.8)} \\ \text{desired dynamics states} \\ \int \mathbf{e} d\tau = \text{integral error terms} \end{bmatrix} \tag{5.16}$$

where:

$$\mathbf{e} = \begin{bmatrix} \theta_1 - \theta_{1d} \\ \theta_2 - \theta_{2d} \end{bmatrix} \tag{5.17}$$

The dynamics matrices are:

$$F(\mathbf{z}) = \begin{bmatrix} A(\mathbf{x}) & 0 & 0 \\ 0 & A_d & 0 \\ C & -C_d & 0 \end{bmatrix} \quad G(\mathbf{z}) = \begin{bmatrix} B(\mathbf{x}) \\ 0 \\ 0 \end{bmatrix} \tag{5.18}$$

where $A(\mathbf{x}), B(\mathbf{x})$ are the same as in equation (5.9), while A_d, the desired dynamics model, is given by:

$$A_d = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\omega_n^2 & -2\zeta\omega_n & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} \tag{5.19}$$

with:

$$\zeta = 1.1 \tag{5.20}$$

$$\omega_n = 1 \tag{5.21}$$

To compute the control input for both the continuous Ricatti design method, and the long-term look-ahead controller, we used a cost function of the form:

$$J = \mathbf{z}^T(t_f)H\mathbf{z}(t_f) + \int_{t_0}^{t_f}(\mathbf{e}_I^T Q_I \mathbf{e}_I + \mathbf{e}^T Q_e \mathbf{e} + \mathbf{x}^T Q_x \mathbf{x} + \rho^2 \mathbf{u}^T \mathbf{u})\mathrm{d}\tau \tag{5.22}$$

$$\stackrel{\text{def}}{=} \mathbf{z}^T(t_f)H\mathbf{z}(t_f) + \int_{t_0}^{t_f}(\mathbf{z}^T Q \mathbf{z} + \rho^2 \mathbf{u}^T \mathbf{u})\mathrm{d}\tau \tag{5.23}$$

$$\tag{5.24}$$

In both cases we also used the same parameters Q_I, Q_e, Q_x and ρ. The numerical values were:

$$Q_I = \begin{bmatrix} 5000 & 0 \\ 0 & 100 \end{bmatrix} \tag{5.25}$$

$$Q_e = \begin{bmatrix} 1 \times 10^6 & 0 \\ 0 & 1 \times 10^5 \end{bmatrix} \tag{5.26}$$

$$Q_x = \mathrm{diag}(\begin{bmatrix} 0 & 100 & 0 & 10 & 0 & 100 & 0 & 10 \end{bmatrix}) \tag{5.27}$$

$$\rho = 1 \times 10^{-4} \tag{5.28}$$

In order to accentuate the nonlinear nature of the system and demonstrate the capability of the control methods used, we chose a reference trajectory which would cause the system to perform a maneuver which we do not expect to be executed in reality. The desired maneuver (see figure 5-2) starts with the manipulator at a stationary position with the arm stretched out (figure 5-2 A). Then the outboard link rotates through 360^o while the inner link rotates through 90^o (figure 5-2 B) until we reach a terminal condition where the manipulator is again fully stretched out but now in a direction 90^o relative to its original orientation (figure 5-2 C).

Controller Based on Linearized Dynamics

To demonstrate the nonlinear nature of the problem we first examine the closed loop response resulting from a linear design. Using the cost function and controller structure described above we found a feedback law by solving an infinite time ($t_f \rightarrow \infty$) linear quadratic regulator problem using a linearized dynamics model for the system. Figure 5-3 shows the resulting unstable behavior.

Continuous Ricatti Design

Figure 5-4 shows the well behaved responses we get when we use a continuous Ricatti design method to control the flexible manipulator. The control input was computed using the system dynamics model and cost function described above. In order to reduce the amount of computation we used the method discussed in Section 3.5, i.e. we used the feedback gain:

$$K(\mathbf{z}) = \frac{1}{\rho^2} R^{-1} G^T(\mathbf{z}) P_{old} \tag{5.29}$$

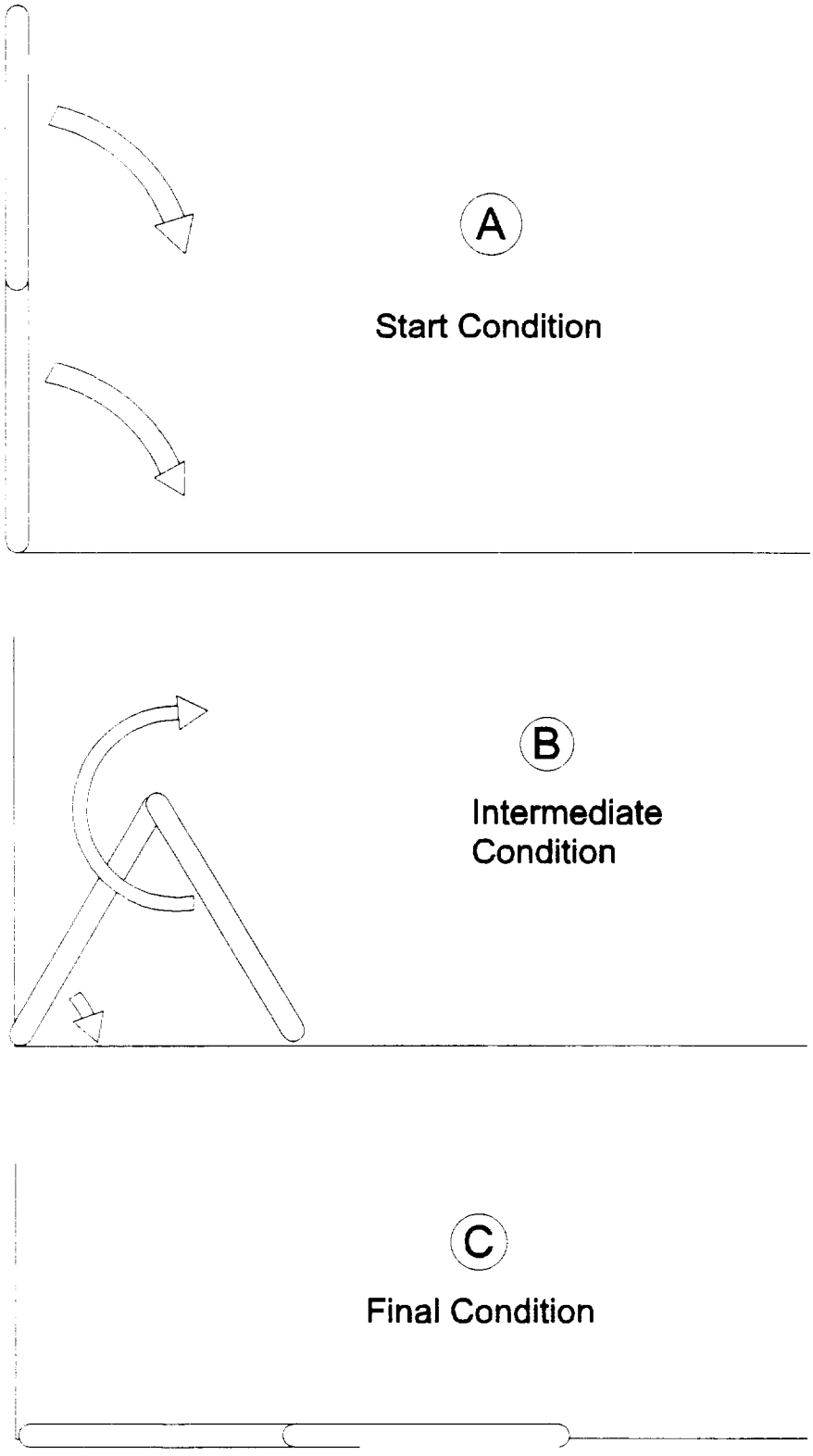


Figure 5-2: Flexible Manipulator Maneuver

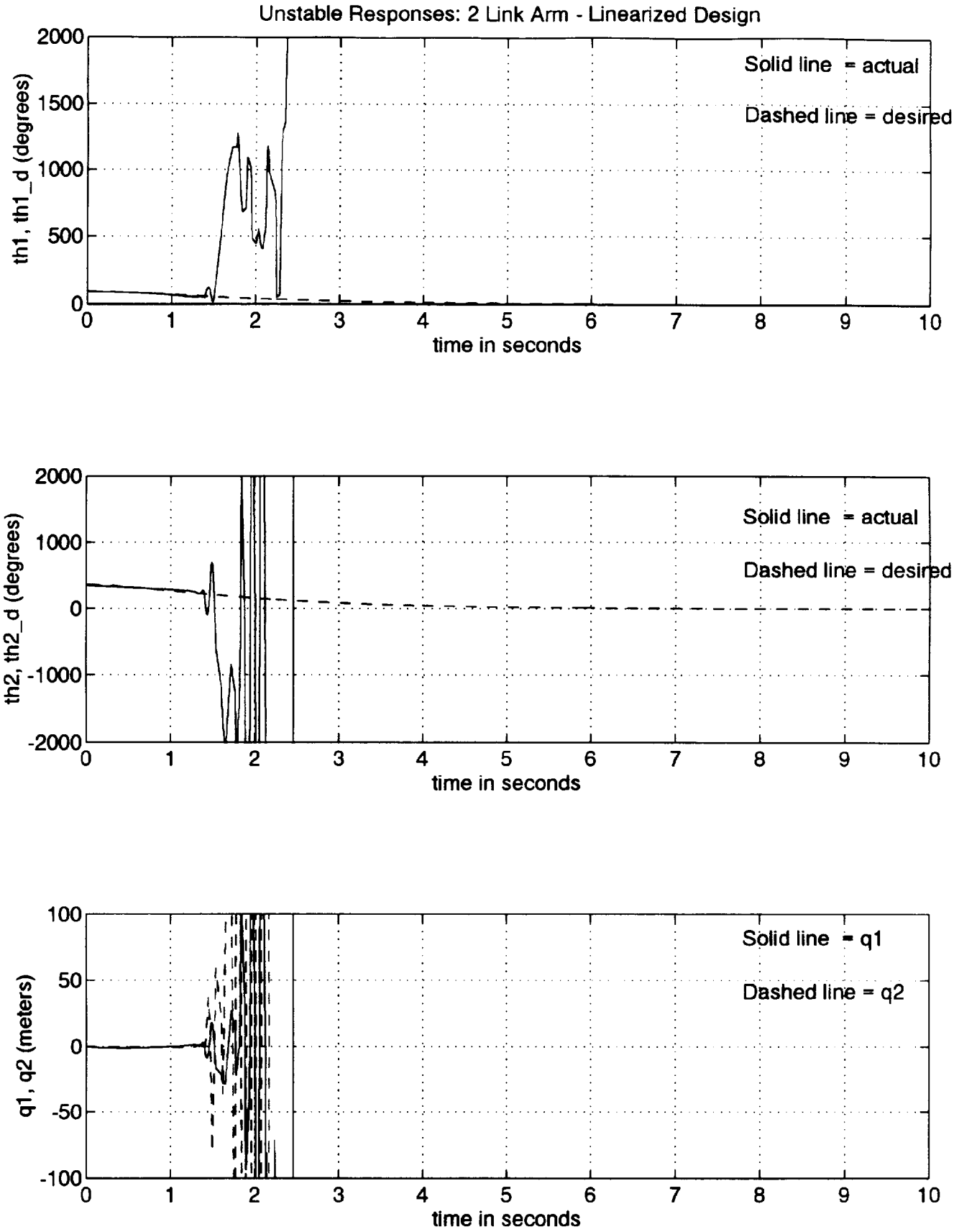


Figure 5-3: Two Link Flexible Manipulator Responses – Linearized Design

where P_{old} was found by re-solving the steady state Ricatti equation whenever the residual:

$$\Delta(\mathbf{z}) = Q + P_{old}F(\mathbf{z}) + F^T(\mathbf{z})P_{old} - \frac{1}{\rho^2}P_{old}G(\mathbf{z})R^{-1}G^T(\mathbf{z})P_{old} \tag{5.30}$$

became large enough that:

$$\frac{\|\Delta(\mathbf{x})\|_2}{\|P_{old}\|_2} > 0.01 \tag{5.31}$$

We see that in this case the continuous Ricatti design controller is able to cope with the nonlinear nature of the dynamics.

Long-term Look-ahead Control

Finally we applied the long-term look-ahead control method to this example. The control input was found as described in Section 4.5, i.e.

- The expected trajectory was estimated by computing the response of the linear time invariant system:

$$\dot{\hat{\mathbf{z}}} = \left(F(\mathbf{z}_0) - \frac{1}{\rho^2}G(\mathbf{z}_0)R^{-1}G^T(\mathbf{z}_0)P_0\right)\hat{\mathbf{z}} \tag{5.32}$$

  where $\mathbf{z}_0$ was the initial condition of the system.

- The feedback gain was found by integrating the time-varying Ricatti equation:

$$-\frac{\partial P(\tau,t_f)}{\partial \tau} = Q + P(\tau,t_f)F(\hat{\mathbf{z}}(\tau)) + F^T(\hat{\mathbf{z}}(\tau))P(\tau,t_f) \tag{5.33}$$

$$-\frac{1}{\rho^2}P(\tau,t_f)G(\hat{\mathbf{z}}(\tau))R^{-1}G^T(\hat{\mathbf{z}}(\tau))P(\tau,t_f) \tag{5.34}$$

  backwards in time. The terminal condition used as a starting point was:

$$P(t_f,t_f) = H = \bar{P} \tag{5.35}$$

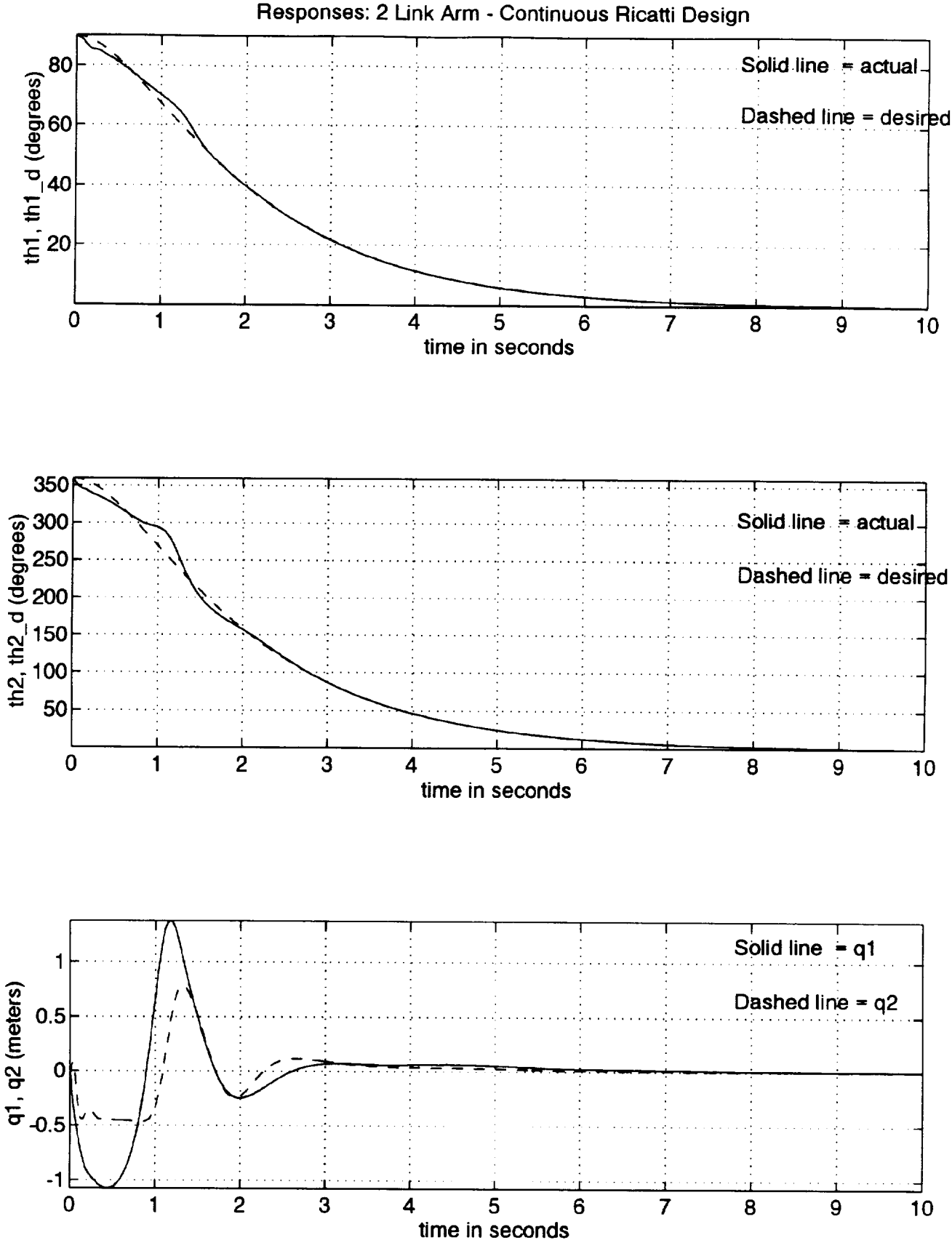


Figure 5-4: Two Link Flexible Manipulator Responses - Continuous Ricatti Design

where $\bar{P}$ was found by solving the steady state Ricatti equation:

$$0 = Q + \bar{P}F(\mathbf{0}) + F^T(\mathbf{0})\bar{P} - \frac{1}{\rho^2}\bar{P}G(\mathbf{0})R^{-1}G^T(\mathbf{0})\bar{P} \tag{5.36}$$

Using this value for H ensured a smooth transition to a controller appropriate for $t > t_f$, since

- for $t > t_f$ the system will be close to the terminal equilibrium point,
- and the feedback gain:

$$K = \frac{1}{\rho^2}R^{-1}G^T(\mathbf{0})H \tag{5.37}$$

will be appropriate for the system dynamics linearized around the equilibrium point.

5.4 Summary

In this Chapter we used both the continuous Ricatti design method of Chapter 4, and the long-term look-ahead control method of Section 3 to obtain controllers for a complex nonlinear system, i.e. a two-link flexible manipulator.

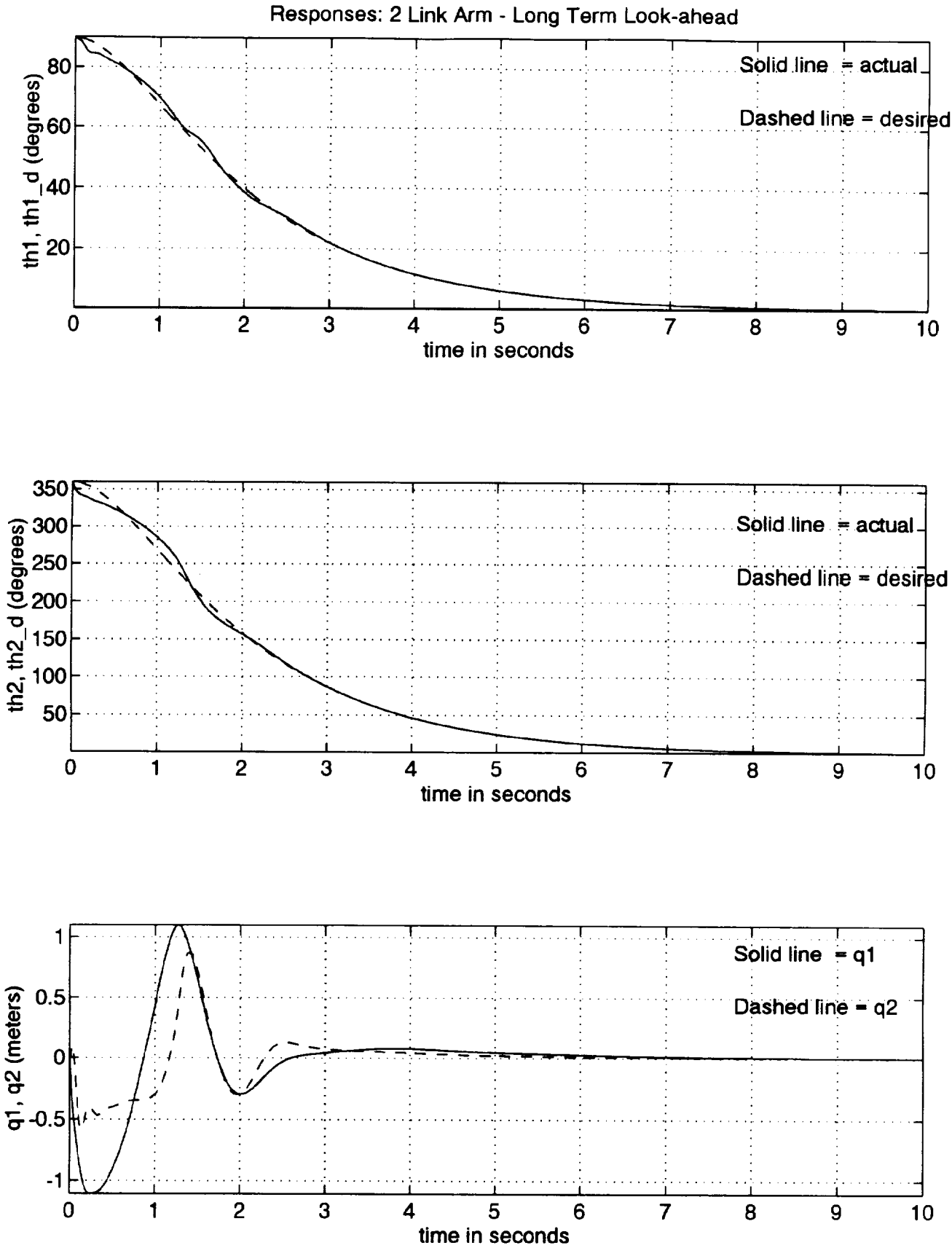


Figure 5-5: Two Link Flexible Manipulator Responses – Long-term Look-ahead

Chapter 6

Conclusions

In this thesis we examined methods to synthesize controllers for nonlinear systems. Our approach was to exploit the fact that nonlinear systems of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u} \tag{6.1}$$

could be expressed in quasilinear form, viz:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{6.2}$$

provided that $\mathbf{f}$ is continuously differentiable [63]. The control methods we developed were generic since a wide range of systems could be expressed in the quasilinear form of equation 6.2. Furthermore the methods allow the designer to conveniently modify the system's response by adjusting parameters of a cost function. The methods however do *not* guarantee global closed loop stability.

We developed two classes of control methods:

- *zero-look-ahead control*, i.e. control laws that determine the control input based only on the current values of $A(\mathbf{x}), B(\mathbf{x})$.

- *controllers with look-ahead*, i.e. control laws that use estimates of the future behavior of $A(\mathbf{x}), B(\mathbf{x})$ to determine the control input.

The first type of control law was found by continuously solving a matrix Ricatti equation as the system progressed along a state trajectory. We found that by appropriately adjusting the cost function the method yielded useful control laws. Best results were obtained when we cast the problem into a combined tracking/regulation problem (see Section 3.2.3) and used very small weights on the control input. With regard to stability we found that for this method we could guarantee local stability only, but motivated our expectation that the system would be stable for a larger range of initial conditions.

The controllers of the second type used the similarity between quasilinear systems and linear time varying systems to find approximate solutions to optimal control type problems. Insofar as the control inputs determined in this way were actually optimal, the closed loop systems would be stable.

Our main conclusion is that the methods we developed provide a useful alternative which control engineers can use to obtain control laws for systems that exhibit significant nonlinearity. However, it also became clear to us that trying to develop control methods that are "generic", while at the same time guaranteeing global stability provides a significant challenge. A more useful approach might be to study nonlinear problems on an "individual" basis so as to exploit the "individual physics" of each problem to develop control laws (see e.g. [56]). For instance, it may be possible to obtain a controller that performs well for the two-link flexible manipulator example we considered, by using a feedback law that is "dissipative". This can be done using control torques that are proportional to the joint angles and angular rates. The effect would be the equivalent of having a spring and dashpot attached to each joint. Since the spring and dashpot combination would be dissipating energy while the system is in motion, and would not be adding energy to the system, we would expect the closed loop system to be stable and eventually settle.

6.1 Future Work

A key issue regarding any control system is closed loop stability. Ideally a design procedure will guarantee (a priori) that the closed loop system will be stable. If this is not the case, the stability of the closed loop system has to be determined as part of an iterative design cycle. In this context Zubov's method (see [21] and Section 3.4.2) provides a powerful method to assess the size of the stability domain for a given design. The results we had show promise, but further work is required to better understand the numerical conditioning of Zubov's equation 3.172.

In Section 2.3.1 we noted that the system:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} \tag{6.3}$$

would be stable provided that the eigenvalues of $A(\mathbf{x})$ were stable, and that the rate of change of the eigenstructure was small enough. This raises the possibility of finding stabilizing controllers for systems:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \tag{6.4}$$

by using an eigenstructure assignment strategy [], which has the aim to keep the eigenstructure as constant as possible. Note that the eigenstructure assignment strategy also provides the designer with the capability to adjust system responses, see e.g. [].

Theorem 2.3.1 provides another avenue for obtaining stabilizing controllers for quasi-linear systems — the goal would be to obtain a closed loop system matrix with diagonal entries that are negative and dominate the off-diagonal elements. For example, we may assume that the control input is given by a feedback law:

$$\mathbf{u} = -K(\mathbf{x})\mathbf{x} \tag{6.5}$$

This will result in the closed loop system:

$$\dot{\mathbf{x}} = A_{cl}(\mathbf{x})\mathbf{x} \tag{6.6}$$

$$= (A(\mathbf{x}) - B(\mathbf{x})K(\mathbf{x}))\mathbf{x} \tag{6.7}$$

A design procedure could then be:

- Find K which minimizes $\|E\|$, where:

$$E = A - BK - \begin{bmatrix} d_1 & & 0 \\ & \ddots & \\ 0 & & d_n \end{bmatrix} \tag{6.8}$$

  This optimization process could be done analytically, resulting in expressions for the optimum values of $K(d_i)$ and $E(d_i)$.

- The next step, which could be done numerically, is to minimize $E(d_i)$ so as to make the diagonal elements of the closed loop system matrix as negative as possible. Theorem 2.3.1 gives the precise conditions needed for this procedure to result in a stable system.

Finally we note that the optimal control approach has desirable features such being "generic" and providing guaranteed stability and robustness. It also allows the designer to conveniently adjust system responses via a choice of cost function. These attractive features should encourage further research in this direction. To make this approach more pratical, it will be necessary to find methods that:

- efficiently solve optimal control problems and provide the answers in terms of a state feedback law (for example by solving the HJB partial differential equation).

- efficiently store state feedback laws found by numerical means.

Appendix A

Modeling and Automatic Generation of Equations of Motion using Maple

A.1 Overview

This document describes a set of routines useful for deriving equations of motion for planar flexible manipulators using the symbolic mathematics software package Maple.

The following can be modelled:

- Planar motion of a flexible manipulator (subject to the base being fixed relative to inertial space.)
- A maximum of 9 links (this restriction can easily be removed)
- Inertial or relative coordinates can be used (see Section A.2.2)
- Each link can be made rigid or flexible, with the following restrictions:

- A Bernoulli-Euler beam model assumed.
- An assumed modes method is used with user supplied mode shape functions.

- Fore-shortening effects (stiffening) can be included. (See Section A.5)
- With regard to the joints the following can be modelled.
  - Joint masses can be specified for each joint.
  - The effects of joint inertias are not included.
  - Flexibility in a joint is modelled as a simple torsional spring (the joint can be made rigid).

A.2 Dynamics Model

In this section we describe the derivation of the equations of motion that were implemented in the routines.

Consider a planar manipulator as seen in Figure A-1 with axis systems as described in Section A.2.2

We can derive the equations of motion using Lagrange's equations, which are:

$$\frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_i}) - \frac{\partial L}{\partial q_i} = Q_i$$

where:

q_i = ith generalised coordinate

Q_i = ith generalised force

T = kinetic energy

V = potential energy

L = $T - V$

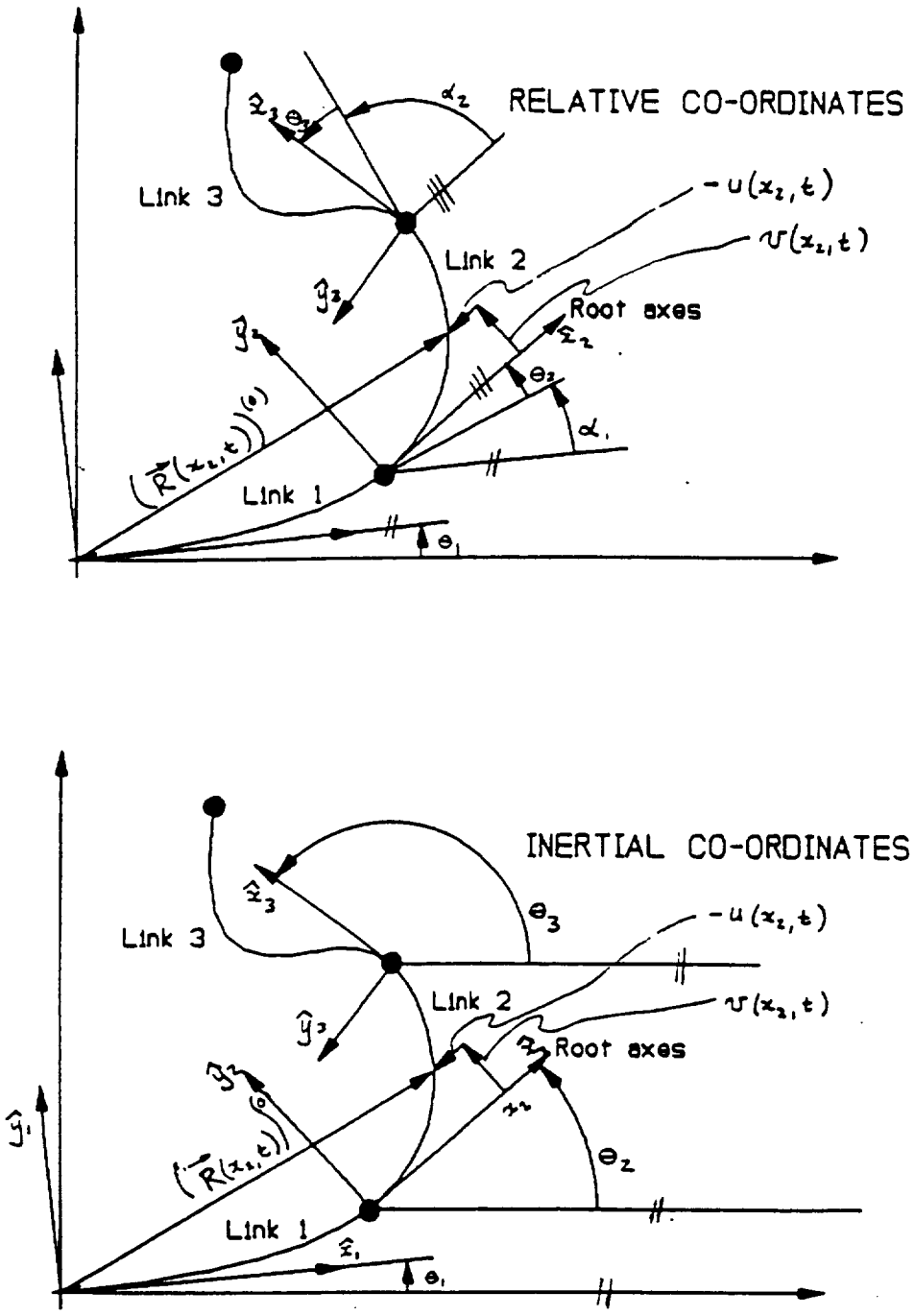


Figure A-1: Coordinate Systems for General Planar Flexible Manipulator

Note that if we can find an expression for the kinetic and potential energy in the form:

$$T = \dot{\mathbf{q}}^T H \dot{\mathbf{q}}$$

$$V = \mathbf{q}^T K \mathbf{q}$$

where:

$\mathbf{q}^T = [q_1\ q_2 \ldots q_n]$ is a vector of generalized coordinates

H = a generalised inertia matrix

K = a generalised stiffness matrix

Lagranges equations can be expressed as:

$$H\ddot{\mathbf{q}} + \dot{H}\dot{\mathbf{q}} - \nabla T + K\mathbf{q} = \mathbf{Q}$$

where:

$\dot{H} = \frac{d}{dt}(H(\mathbf{q}))$ is the time derivative of the generalised inertia matrix

$Q^T = [Q_1\ Q_2 \ldots Q_n]$ is a vector of generalised forces

$\nabla T = [\frac{\partial T}{\partial q_1}, \frac{\partial T}{\partial q_2}, \ldots, \frac{\partial T}{\partial q_n}]$

In the next few sections we will find expressions for the kinetic and potential energy of a flexible manipulator.

A.2.1 Notation

In general symbols will have the following meaning:

- A subscript, or superscript, i, will associate a variable with the ith link.
- N denotes the number of links of the manipulator.
- L_i denotes the nominal length of link i.

A.2.2 Axis systems

Depending on whether we use *inertial* or *relative* coordinates the following sets of axis systems will be used (see also Figures A-1 and A-2).

Axis Systems for Relative Coordinates

We have:

- An *inertial axis system* $\hat{\mathbf{x}}_0, \hat{\mathbf{y}}_0$ fixed at the base of the arm.
- A *rotor axis system* $\hat{\mathbf{x}}_{r_i}$, $\hat{\mathbf{y}}_{r_i}$ for each link. This axis system is associated with the rotor of the torque motor at the root of the ith link (see Figures A-1 and A-2). This axis system is only of importance if we are including flexibility in the joints (see also section A.2.4). The orientation of the rotor axis system can be specified by an an angle, ψ_i, relative to the line tangent to the tip of the previous link.
- A *root axis system* $\hat{\mathbf{x}}_i$, $\hat{\mathbf{y}}_i$ for each link associated with the root of the link. In this case the orientation of the axis system is measured relative to the line tangent to the tip of the previous link. The angle θ_i for this case would typically be the angle measured by a joint angle sensor between the previous link and the output from a gearbox. The deflection of a link is measured relative to the *root axis system.*

Relative coordinates would typically be used when:

- We want state variables which are directly measured (i.e. joint angles).
- A direct drive arm is being modelled.

However *relative coordinates* usually result in more complex equations.

Axis Systems for Inertial Coordinates

We have:

- An *inertial axis system* $\hat{\mathbf{x}}_0, \hat{\mathbf{y}}_0$ fixed at the base of the arm as before.
- A *rotor axis system* $\hat{\mathbf{x}}_{r_i}$, $\hat{\mathbf{y}}_{r_i}$ for each link, similar to the case for *relative coordinates*. The difference is that now the orientation of the axis system is measured relative to an inertial reference (see Figure A-1).
- A *root axis system* $\hat{\mathbf{x}}_i$, $\hat{\mathbf{y}}_i$ for each link associated with the root of the link. In this case the angle θ_i specifies the orientation of the *root axis system* relative to the *inertial axis system*. The deflection of a link is measured relative to the *root axis system*.

Inertial coordinates would typically be used when:

- An indirect drive manipulator is being modelled, i.e. the joints are driven via some drive mechanism at the base of the manipulator.
- We want simpler equations of motion.

However using *inertial coordinates* results in equations of motion where the state variables are not directly measured.

Notes

The deflection of the link is measured relative to the *root axis system*. By suitable choice of the mode shape functions, the $\hat{\mathbf{x}}_i$ axis could be, e.g.:

- tangent to the root of the link, e.g., by using clamped-free cantilever mode shapes.

- such that the extension of the $\hat{\mathbf{x}}_i$ axis joins the root and tip of the link, e.g., by using mode shapes of the form:

$$\phi(x_i) = sin(n\pi\frac{x_i}{L_i}) \qquad n = \pm 1, 2, 3 \ldots$$

where:

L_i = nominal length of the link

x_i = distance measured along $\hat{\mathbf{x}}_i$

Note that when joint flexibility is taken into account the *twist angle* between the "input and output of the gearbox" will be given by:

$$\theta_{\text{twist}} = \psi_i - \theta_i$$

for both *inertial* and *relative* coordinates.

A.2.3 Kinetic energy

Kinetic Energy of the Link (excluding root and tip masses)

Consider an arbitrary link i and an arbitrary "slice" of this link. The "slice"is located at a distance x_i along the $\hat{\mathbf{x}}_i$ axis when the link is in the undeflected state. We will first find an expression for a position vector from the inertial axis system to this slice and then take time derivatives etc., to find the kinetic energy associated with the slice. The kinetic energy of each link can then be found by integrating over the length of the link.

To keep track of the ferent axis systems we will use the following notation. A vector, $\vec{R}$, will be written as:

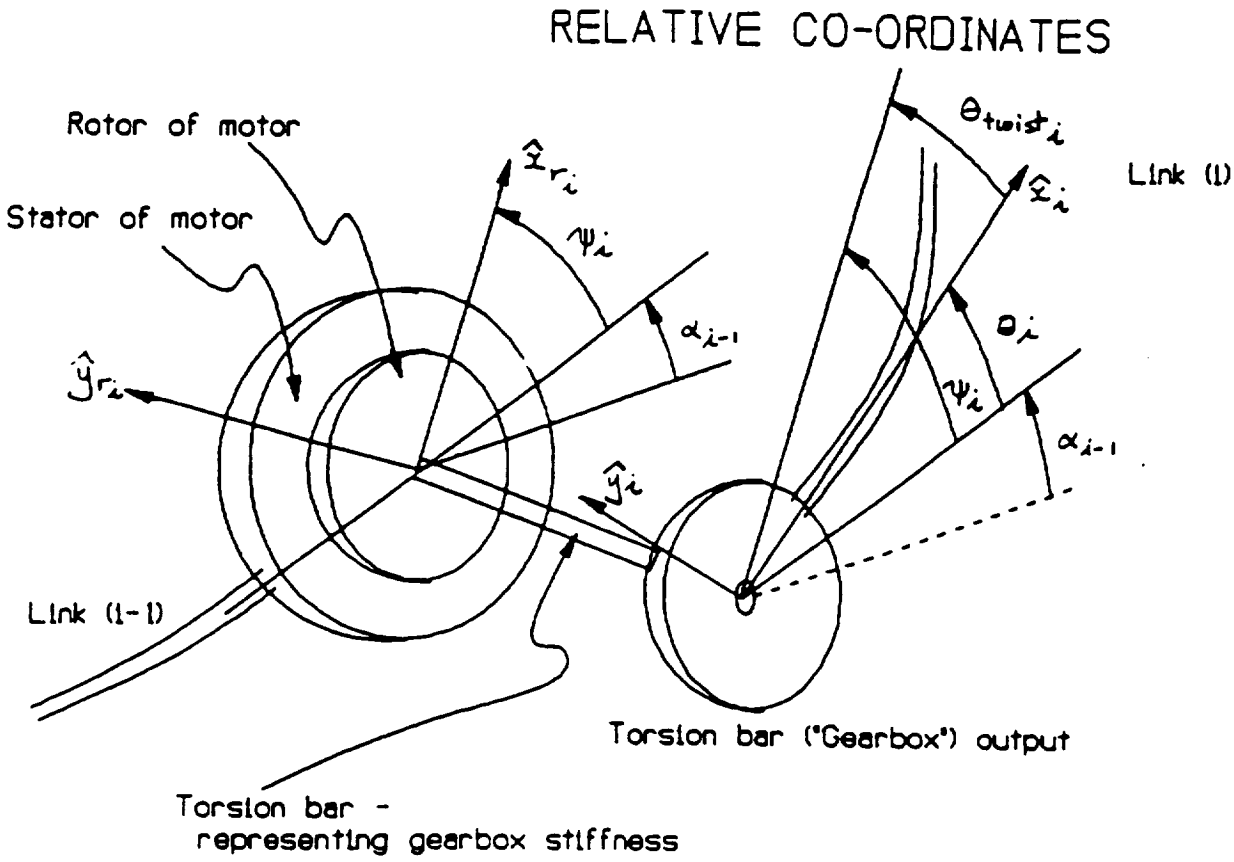


Figure A-2: Model of joint at the root of link i

$$(\vec{R})^{(i)}$$

The superscript (i) denotes that the vector is measured *relative* to axis system $\hat{\mathbf{x}}_i$, $\hat{\mathbf{y}}_i$, i.e. when we are dealing with a postion vector, the vector is measured from the origin of the axis system indicated by the superscript. We will also, unless stated otherwise, assume that the vector is *coordinatised* in its natural axis system, i.e. if we write the vector $(\vec{R}(x_i))^{(i)}$ as a column of numbers, the numbers will give the components of $(\vec{R}(x_i))^{(i)}$ in the $\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i$ frame.

First find the position vector $\vec{R}(x_i)^{(i)}$ from the origin of axis system $\hat{\mathbf{x}}_i$,$\hat{\mathbf{y}}_i$ to "slice"x_i: When link i deflects, the slice x_i moves a distance $u(x_i, t)$ parallel to the $\hat{\mathbf{x}}_i$ axis and a distance $v(x_i, t)$ parallel to the $\hat{\mathbf{y}}_i$ axis (directions for positive movement are shown in Figure A-1) (the variable t indicates time dependence). Hence:

$$(\vec{R}(x_i, t))^{(i)} = (x_i + u(x_i, t))\hat{\mathbf{x}}_i + (v(x_i, t))\hat{\mathbf{y}}_i$$

or in matrix form:

$$(\vec{R}(x_i))^{(i)} = \begin{bmatrix} x_i + u(x_i, t) \\ v(x_i, t) \end{bmatrix}$$

Since we are using an assumed modes method we have:

$$v(x_i, t) = \sum_{j=1}^{n_i} \phi_{ij}(x) q_{ij}(t)$$

where: $\phi_{ij}(x)$ are the assumed mode shapes for link i.

q_{ij} are the generalized coordinates for the deflection of link i.

Foreshortening effects can be taken into account as shown in Section A.5. When foreshortening effects are taken into account we have:

$$u(x_i,t) = -\frac{1}{2}\int_0^{x_i}\left(\frac{\partial v(s_i,t)}{\partial s_i}\right)^2 ds_i$$

When we ignore foreshortening we set:

$$u(x_i,t) = 0$$

To transform the vector $(\vec{R}(x_i))^{(i)}$ to inertial axis systems we will use the so-called "homogenous rotation matrices" as described in [3]. To do this we first augment our position vector to get:

$$(X(x_i,t))^{(i)} = \begin{bmatrix} x_i + u(x_i,t) \\ v(x_i) \\ 1 \end{bmatrix}$$

(Note that the superscript has a similar meaning as before). The augmented position vector to "slice" x_i measured from the origin of axis system $\hat{\mathbf{x}}_{i-1}$,$\hat{\mathbf{y}}_{i-1}$ and coordinatised in axis system $\hat{\mathbf{x}}_{i-1}$,$\hat{\mathbf{y}}_{i-1}$ is then given by:

$$X(x_i,t)^{(i-1)} = A_i^{i-1}X(x_i,t)^{(i)}$$

where:

$$A_i^{i-1} = \begin{bmatrix} C_i^{i-1} & (\vec{R}(L_{i-1},t))^{(i-1)} \\ 0 \quad 0 & 1 \end{bmatrix}$$

and

$$\vec{R}(L_{i-1},t)^{(i-1)} = \begin{bmatrix} L_{i-1} + u(L_{i-1},t) \\ v(L_{i-1},t) \end{bmatrix}$$

$$C_i^{i-1} = \begin{bmatrix} \cos(\theta_{rel_i}) & -\sin(\theta_{rel_i}) \\ \sin(\theta_{rel_i}) & \cos(\theta_{rel_i}) \end{bmatrix}$$

and θ_{rel_i} is the relative angle between axis systems $\hat{\mathbf{x}}_i$,$\hat{\mathbf{y}}_i$ and $\hat{\mathbf{x}}_{i-1}$, $\hat{\mathbf{y}}_{i-1}$.

We get the following expressions for θ_{rel_i}:

- For relative coordinates:(see Figure A-1)

$$\theta_{\mathrm{rel}_i} = \theta_i + \alpha_{i-1}$$

  where α_{i-1} is the slope (in radians) of link $i-1$ at its tip (measured relative to $\hat{\mathbf{x}}_{i-1}, \hat{\mathbf{y}}_{i-1}$).

- For inertial coordinates:

$$\theta_{rel_i} = \theta_i - \theta_{i-1}$$

Assuming small deflections we have:

$$\alpha_i = \frac{\partial v(x_i, t)}{\partial x_i}|_{L_i}$$

We can then transform the augmented vector $X(x_i, t)^{(i)}$ back to the inertial coordinate system by repeated application of the relation above. We finally get:

$$X(x_i, t)^{(0)} = A_1^0 A_2^1 \ldots A_i^{i-1} X(x_i, t)^{(i)}$$

and extract $(\vec{R}(x_i, t))^{(0)}$ (position vector from inertial axes to "slice"x_i) as the first two components of $(X(x_i))^{(0)}$.

Find an expression for the velocity of the "slice": Small motions of slice x_i are related to small changes in the generalised coordinates via the Jacobian matrix,

i.e.:

$$d\vec{R}(x_i,t)^{(0)} = [J(x_i,t)]d\mathbf{q}$$

where:

$$J(x_i,t) = \left[\frac{\partial \vec{R}(x_i,t)^{(0)}}{\partial \mathbf{q}}\right]$$

and $\mathbf{q}$ is now a vector of generalized coordinates for the manipulator. In the most general case where joint flexibility is included we have:

$$\mathbf{q} = \begin{bmatrix} \psi_1 \\ \theta_1 \\ q_{11} \\ q_{12} \\ q_{13} \\ \vdots \\ \psi_2 \\ \theta_2 \\ q_{21} \\ q_{22} \\ \vdots \\ \psi_N \\ \theta_N \\ q_{N1} \\ q_{N2} \end{bmatrix}$$

Dividing both sides by dt we get an expression for the velocity of slice x_i in inertial coordinates viz:

$$\dot{\vec{R}}(x_i,t)^{(0)} = J(x_i,t)\dot{\mathbf{q}}$$

Find the kinetic energy of the ith link: We do this by summing the contributions of all the slices in the link, i.e., integrating over the length of the link. (It is best to think of x_i as a label for the specific slice on the ith link - thus we integrate from 0 to L_i). The kinetic energy of link i is given by:

$$T_i = \frac{1}{2}\int_0^{L_i} \rho A_i \dot{\mathbf{q}}^T J^T(x_i,t) J(x_i,t)\dot{\mathbf{q}} dx_i$$

where:

ρA_i is the mass per unit length of link i

L_i is the length of link i

$J(x_i,t)$ is the Jacobian of the position vector $\vec{R}(x_i,t)^{(0)}$ indicated above.

Kinetic Energy of a Point Mass at the tip and root of a link

Using the same notation as above we see that motion of a point mass at the tip of link i can be found by setting:

$$x_i = L_i$$

and

$$\rho_A dx_i = m_{L_i}$$

where m_{L_i} is the point mass at the tip of link i.

So, using the same equations as in the previous section, the expression for the kinetic energy of a point mass at the tip of link i is:

$$T_{\text{tip}_i} = \frac{1}{2} m_{L_i} \dot{\mathbf{q}}^T J^T(L_i) J(L_i) \dot{\mathbf{q}}$$

Similarly, the kinetic energy of a point mass at the root of link i is:

$$T_{\text{root}_i} = \frac{1}{2} m_{0_i} \dot{\mathbf{q}}^T J^T(L_{i-1}) J(L_{i-1}) \dot{\mathbf{q}}$$

Total kinetic energy

Adding the contributions of each link, the total kinetic energy for the system is given by:

$$T = \sum_{i=1}^{N} (T_i + T_{\text{tip}_i} + T_{\text{root}_i})$$

We have neglected:

- The effects of rotary inertia of the beam slices
- The effects of rotary inertia at the joints

A.2.4 Potential Energy

Potential Energy Associated with the bending of a link

For a simple Bernoulli-Euler beam model the potential energy associated with the bending of link i is given by:

$$V_i = \frac{1}{2} \int_0^{L_i} EI_i \left(\frac{\partial^2 v(x_i, t)}{\partial x_i^2} \right)^2 dx_i$$

where EI_i is the bending stiffness of link i.

Again, using the assumed modes approach we get:

$$V_i = \frac{1}{2}\int_0^{L_i} EI_i \left(\sum_{j=1}^{n_i} \phi''_{ij}(x_i) q_{ij}(t)\right)^2 dx_i$$

which can be rewritten as:

$$V_i = \tilde{q}_i{}^T \left[\int_0^{L_i} EI_i \left[\Phi''(x_i,t)\right] dx_i\right] \tilde{q}_i$$

where:

$$\tilde{q}_i{}^T = \begin{bmatrix} \psi_i & \theta_i & q_{i1} & \cdots & q_{in_i} \end{bmatrix}$$

$$[\Phi''(x_i,t)] = \begin{bmatrix} \phi''_{i1}\phi''_{i1} & \phi''_{i1}\phi''_{i2} & \cdots & \phi''_{i1}\phi''_{in_i} \\ \phi''_{i2}\phi''_{i1} & \phi''_{i2}\phi''_{i2} & \cdots & \phi''_{i2}\phi''_{in_i} \\ \vdots & & & \\ \phi''_{in_i}\phi''_{i1} & \phi''_{in_i}\phi''_{i2} & & \phi''_{in_i}\phi''_{in_i} \end{bmatrix}$$

Potential Energy Associated with joint flexibillity

For joint i (joint i appears at the root of link i) we find:

$$V_{joint_i} = \frac{1}{2} k_i (\psi_i - \theta_i)^2$$

where k_i is the joint stiffness of the ith joint. Note that this relation holds for both *inertial* and *relative* coordinates.

Total Potential Energy

The potential energy for link i and joint i using relative coordinates can then be written in matrix form as:

$$V_i + V_{\text{joint}_i} = \tilde{q}_i{}^T K_i \tilde{q}_i$$

where:

$$K_i = \begin{bmatrix} k_i & -k_i & 0 & \dots & 0 \\ -k_i & k_i & 0 & \dots & 0 \\ 0 & 0 & \int_0^{L_i} EI_i \phi''_{i1} \phi''_{i1} dx_i & \dots & \int_0^{L_i} EI_i \phi''_{i1} \phi''_{in_i} dx_i \\ \vdots & & & & \\ 0 & 0 & \int_0^{L_i} EI_i \phi''_{in_i} \phi''_{i1} dx_i & & \int_0^{L_i} EI_i \phi''_{in_i} \phi''_{in_i} dx_i \end{bmatrix}$$

Then collect the total Potential energy for the system as:

$$V_{tot} = \frac{1}{2}\mathbf{q}^T \begin{bmatrix} K_1 & 0\dots0 & 0\dots0 & \dots & 0\dots0 \\ 0\dots0 & K_2 & 0\dots0 & \dots & 0\dots0 \\ \vdots & & & & \\ 0\dots0 & 0\dots0 & 0\dots0 & 0\dots0 & K_N \end{bmatrix} \mathbf{q}$$

Notes:

- The effects of gravity have been neglected.
- The effects of shear deformation have been neglected.

A.2.5 Equations of Motion

Using the expressions for kinetic and potential energy we find the equations of motion are as indicated before:

$$H(\mathbf{q})\ddot{\mathbf{q}} + \dot{H}\dot{\mathbf{q}} - \nabla T + K\mathbf{q} = \mathbf{Q}$$

It is often convenient to collect the second and third terms in the equation above as:

$$C(\mathbf{q}, \dot{\mathbf{q}}) = \dot{H}\dot{\mathbf{q}} - \nabla T$$

A.3 Using the Routines

The routines calculate the following terms in Lagrange's equations for a flexible manipulator:

$$H(\mathbf{q}), \qquad \dot{H}\dot{\mathbf{q}}, \qquad \nabla T, \qquad K$$

as well as:

$$C(\mathbf{q}, \dot{\mathbf{q}})$$

The characteristics of each manipulator, for which we want to get equations of motion, are specified by filling in the details in a template file such as *"template.mpl"* (see Section A.6). Note the following conventions which must be adhered to:

- Upper and lower case is important.
- The names of the generalised coordinates must be specified analogous to those shown in *"template.mpl"*, i.e.:
  - For the rotor angles we must use:
    `PSI1, PSI2, ....`
    where the digits specify the link numbers. If joint flexibility is not modelled the rotor angles can be omitted totally.

- For the angles at the roots of the links we use:

  `TH1, TH2, ....`

- For the deflection coordinates we must use:

  `Q11, Q12,..., Q21,Q22, .....`

  where the first digit in each name specifies the link number, and the second digit in each deflection coordinate specifies the mode number for the given link.

- The names of the mode shapes must be of the form:

  `phi11, phi12,.....,phi21, phi22,.....`

  where the first digit in each name specifies the link number, and the second digit in each shape specifies the mode number for the given link.

- The names of the mode shapes must be in lower case as shown.

- Mode shape functions must be specified in Maple operator form, e.g.:

  `phi11:= < sin(Pi*x/L1) | x > ;`

To run the routines one must first start Maple with a "clean slate" (e.g. by quitting Maple and then starting it again). The user should then specify the file which contains the specifications of the system for which the the equations of motion are required. This is done by typing, e.g.:

```
> modelname:=''template.mpl'';
```

(note that the user can create a specification file with an arbitrary name as long as it contains the relevant data in the same form as shown in *"template.mpl"*).

To run the routines the user then types:

```
> read 'armmain.mpl';
```

This causes the module *armmain.mpl* to be executed, which reads the data from *"template.mpl"* and does minor error checking. It also defines some variables, i.e.

iner, *stiff* and *lagran* which associates the correct file names with the modules which calculate the different terms in Lagranges equations. The user can modify *arm-main.mpl* to reflect the proper file names for the user's site.

Next the user types:

```
> read iner;
```

This module calculates an initial form of the inertia matrix $H(\mathbf{q})$. It generally takes the longest to execute.

After the *iner* module has completed the user types:

```
> read stiff;
```

This module calculates the stiffness matrix.

Finally the user types:

```
> read lagaran;
```

This module calculates the final forms of the terms in Lagrange's equations.

Throughout the session the various modules write messages to the screen, indicating their progress to the user. Section A.7 gives an annotated version of a session where the routines were used.

A.4 Description of routines used

The main modules are:

`TEMPLATE.MPL` (or equivalent)	:	Input data specifying the manipulator characteristics. The user has to supply this.
`armmain.mpl`	:	The main routine for the package
`iner?.mpl`	:	Calculates the inertia matrix
`stiff?.mpl`	:	Calculates the stiffness matrix
`lagran.mpl`	:	Calculates the matrices $\dot{H}(\mathbf{q})$ ∇T $C(q, \dot{\mathbf{q}})$ in Lagranges equations

Note:

The "?" denotes digits related to version numbers of the routines. These may change with time.

Subroutine modules are:

`armmsub.mpl`	:	subroutines used by `armmain.mpl`
`inersub?.mpl`	:	subroutines used by `iner?.mpl`
`stiffsub.mpl`	:	subroutines used by `stiff?.mpl`

A.5 Foreshortening

In this section we use a simple geometric argument from [47] to determine the stiffening terms in the equations of motion. Alternative derivations are given in [54].

Assume that the length of each link, measured along the curved path it makes in space, stays fixed, with value, L_i. Thus the projection of the vector from the origin of the axes $\hat{\mathbf{x}}_i\,\hat{\mathbf{y}}_i$ to the tip of link i will, in general not have an $\hat{\mathbf{x}}_i$ axis projection of L_i.

We can find the expression for the motion of the slice labelled x_i, in the $\hat{\mathbf{x}}_i$ direction, by considering an arbitrary element of the link with length Δs (see figure A-3). For the moment we will only consider the $\hat{\mathbf{x}}_i$ axis motion of the tip of this element, relative to its own root. The motion of the root of the element, will be accumulated by the small elements to the "left" of Δs. We see that the tip of the element moves *locally* inwards (due to the local slope), by a distance $\Delta\tilde{u}$. Let the projection of the element, Δs, on the $\hat{\mathbf{x}}_i$ axis be Δx. Then, by Pythagoras' theorem and, using the binomial expansion for the square root we get:

$$\begin{aligned} \Delta s &= (\Delta x^2 + \Delta v^2)^{\frac{1}{2}} && \text{(A.1)} \\ &= (1 + (\frac{\Delta v}{\Delta x})^2)^{\frac{1}{2}} \Delta x && \text{(A.2)} \\ &\simeq (1 + \frac{1}{2}(\frac{\partial v}{\partial x})^2)\Delta x && \text{(A.3)} \end{aligned}$$

so that:

$$\begin{aligned} \Delta\tilde{u} &= \Delta s - \Delta x && \text{(A.4)} \\ &= \frac{1}{2}(\frac{\partial v}{\partial x})^2 && \text{(A.5)} \end{aligned}$$

The cumulative effect of such inward motions is found by integration. This results in the expression:

$$u(x_i, t) = -\frac{1}{2}\int_0^{x_i} \left(\frac{\partial v(s_i, t)}{\partial s_i}\right)^2 ds_i$$

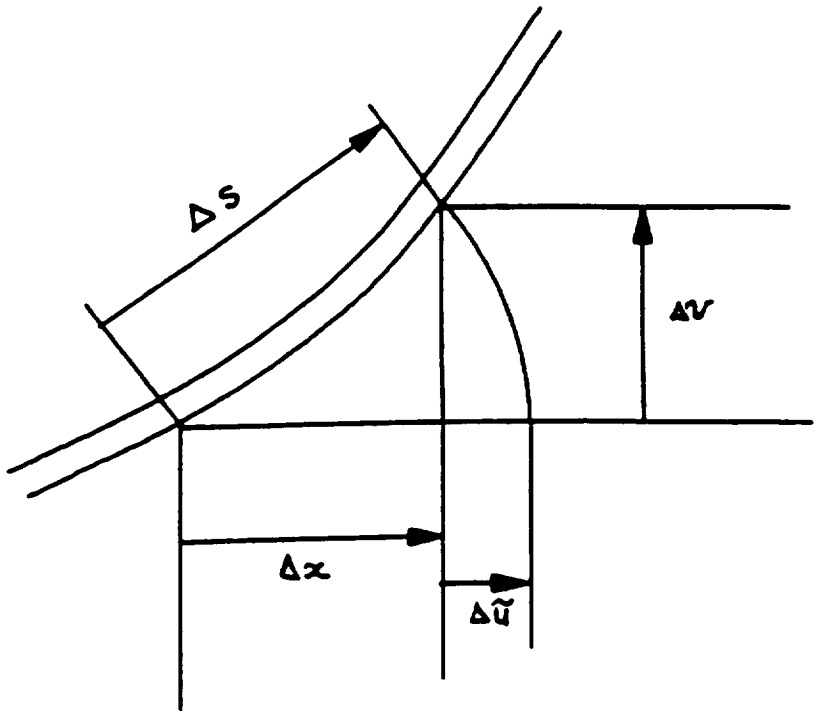


Figure A-3: Foreshortening

A.6 Example of "template.mpl"

```
# AUTHOR: J.A.Coetsee
# 12/10/90: Does integrations late as possible
# Maple v4.3.1 (IBM-PC)
# --------------------
#
#*************************************************************
#  FILL OUT THE FOLLOWING DATA TEMPLATE FOR EACH LINK      *
#*************************************************************
#            |----------------------|
# !! Use the | PRESCRIBED  CAPITAL  | letters for generalised co-ordinates
#            |             LOWERCASE |   "      "  mode shapes
#            |----------------------|
```

```
#    NOTE:
#         ASSUMES mode shapes are orthogonal per link
#                 (see procedure "orthorel" in file "linksub?.mpl")
#
#-------------------------------------------------------------------
#
#  DESCRIPTION:  2 link flexible arm
#                Inertial Co-ordinates
#                Sinusoid mode shapes
#
#  DATE:         11/16/90
#
#-------------------------------------------------------------------
COORDS:= inertial;                          # Co-ordinate system used:
 # "relative":- link orientation THi
 #              measured relative to
 #              tip of previous link
 # "inertial":- link orientation THi
 #              measured relative to
 #              inertial frame
FORESHORTEN:=no;                            # foreshortening effect
#-------------------------------------------------------------------
NL:=2;                                      # Number of links
#-------------------------------------------------------------------
# Data template for LINK 1:
n1:= 1;                                     # Number of modes in the link
qstar1:=[TH1,Q11];                          # List of generalised co-ords. link 1
L1 := L;                                    # Link length
m01:= 0;                                    # Root mass
mL1:= m0;                                   # Tip mass
```

```
k1:= infinity;                          # Joint stiffness
EI1:=EI;                                # Beam stiffness
rhoA1:= rhoA;                           # Beam density per unit length
phi11:= <-sin(Pi*x/L) | x >;                    # Mode shape 1
phi12:= <-sin(2*Pi*x/L) | x >;                  # Mode shape 2

#-----------------------------------------------------------------------
# Data template for LINK 2:
n2:= 1;                                 # Number of modes in the link
qstar2:=[TH2,Q21];                      # List of generalised co-ords. link 2
L2 := l2;                               # Link length
m02:= 0;                                # Root mass
mL2:= m;                                # Tip mass
k2:= infinity;                          # Joint stiffness
EI2:= EI;                               # Beam stiffness
rhoA2:= rhoA;                           # Beam density per unit length
phi21:= <-sin(Pi*x/l2) | x >;           # Mode shape 1
phi22:= <-sin(2*Pi*x/l2) | x >;         # Mode shape 2

#*********************************************************************
```

A.7 Annotated Example of Session

```
    |\^/|
._|\|   |/|_. Copyright (c) 1989 by the University of Waterloo
 \  MAPLE  /  Version 4.3.1VM/80387 - February 1990
 <____ ____>  F1-Help  F2-Search  F3-Quit  F4-Shell
```

```
# First give the name of the file specifying the arm for which we want
# equations of motion

>modelname:='template.mpl';
                          modelname := template.mpl

# Now run the main routine which sets everything up
#    (The routines inform the user of their progress )

>read 'armmain.mpl';
        ------------------------------------------------------------

        |              Main program - Does initialization           |

        ------------------------------------------------------------

Warning: new definition for   trace
                             printlevel := 1

                                 q := []

                              q := [TH1, Q11]

                          q := [TH1, Q11, TH2, Q21]

                                    i := i

                      q := array(1 .. 4,[TH1,Q11,TH2,Q21])
```

```
                    qstarsize := 2

                    qstarsize := 2

                   inertia := iner6.mpl

                  stiffness := stiff.mpl

                  lagrange := lagran.mpl

# Then run the file which calculates the initial form of the generalised
# inertia matrix

>read inertia;
        ------------------------------------------------------------

        |              Calculating the inertia matrix               |

        ------------------------------------------------------------

         ..........................................................

                        Busy with link , 1

         ..........................................................

                          doing v(x)

bytes used=452504, alloc=204800, time=3.430
```

```
doing u(x)

doing A

doing R0

doing Jacobian

doing JJ

doing initial integration of JJ

time for integration of Jacobian is:, .170

busy with Hiroot

time for simplifying Hiroot is, .170

busy with Hitip

adding up the components of the H-matrix

bytes used=854452, alloc=360448, time=6.840
.....

Busy with link , 2

.....

doing v(x)
```

```
                    doing u(x)

                    doing A

                    doing R0

                  doing Jacobian

                    doing JJ

bytes used=1254688, alloc=483328, time=10.190
                  doing initial integration of JJ

              time for integration of Jacobian is:, .110

                    busy with Hiroot

                time for simplifying Hiroot is, 0

                    busy with Hitip

               adding up the components of the H-matrix

          Starting final simplification of H matrix

               - getting constansts outside integral signs

bytes used=1655236, alloc=565248, time=13.650
               - applying orthogonality conditions
```

```
                - doing final integrations with the actual mode-shapes

bytes used=2058144, alloc=565248, time=18.760
bytes used=2458276, alloc=565248, time=24.470
bytes used=2858508, alloc=565248, time=30.180
bytes used=3259148, alloc=565248, time=34.520
bytes used=3659364, alloc=565248, time=38.480
                -doing final collection of terms

            Time for last few simplifications of H is      , 24.280

                                printlevel := 1



# The first routine has just completed
# At this point the user can examine values etc. e.g.

>H[1,1];
                          2                3       2         2          2
          1/2 rhoA Q11  L  + 1/3 rhoA L  + m0 L  + rhoA L  l2 + m L



# Now run the file which calculates the stiffness matrix

>read stiffness;
        ------------------------------------------------------------
```

Calculating the stiffness matrix

.....

Busy with link , 1

.....

bytes used=4059568, alloc=606208, time=41.860

$$\text{K-matrix} = \frac{4}{\pi L^3} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

.....

Busy with link , 2

.....

[0 0 0 0]

C-3

```
                                [                          ]
                                [          4               ]
                                [        Pi                ]
                                [ 0  1/2 ---  0     0      ]
                                [         3                ]
                                [        L                 ]
                  K-matrix  ,   [                          ]
                                [                          ]
                                [                      4   ]
                                [                    Pi    ]
                                [ 0     0     0  1/2 ---   ]
                                [                      3   ]
                                [                    12    ]

                             printlevel := 1

# The second routine has just completed

# Then calculate the terms in Lagrange's equations
>read lagrange;
        ------------------------------------------------------------

        |        Calculating the terms in Lagrange's equations      |

        ------------------------------------------------------------

bytes used=4462016, alloc=606208, time=45.760
         ..........................................................

                      Simplifying the "Coriolis" Terms
```

```
bytes used=4862464, alloc=614400, time=50.370
bytes used=5262596, alloc=614400, time=54.930
bytes used=5662784, alloc=614400, time=59.760
                        Collecting terms in a useful order

bytes used=6063704, alloc=638976, time=64.210
       Combining terms like: sin(a)*cos(b) + sin(b)*cos(a) ---> sin(a+b)

bytes used=6463844, alloc=679936, time=67.890
                                printlevel := 1


# The terms now show time dependence e.g.
>Coriolis[2];
                                  /  d        \2
                          - 1/2 L |---- th1(t)|  rhoA q11(t)
                                  \ dt        /

>H[1,1];
                          2                3       2         2         2
          1/2 rhoA q11(t)  L + 1/3 rhoA L  + m0 L  + rhoA L  l2 + m L

# The results can be saved to a file if need be e.g.

>save HH, Coriolis, K, modelname, 'results.out';

# To end the session

> quit;
```

Appendix B

Quasilinear Equations of motion for Double Pendulum Example

The equations of motion for the double pendulum of Example 2.2.1 are:

$$H(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + K\mathbf{q} + G_{grav}(\mathbf{q})\mathbf{q} = \mathbf{0} \tag{B.1}$$

where:

$$\begin{aligned}
H(1,1) &= m_1 l_1^2 + m_2 l_1^2 + 2 m_2 l_2 l_1 \sin(\theta_1)\sin(\theta_1+\theta_2) + m_2 l_2^2 && \text{(B.2)}\\
&\quad +2 m_2 l_2 l_1 \cos(\theta_1)\cos(\theta_1+\theta_2) && \text{(B.3)}\\
H(1,2) &= m_2 l_2 l_1 \sin(\theta_1)\sin(\theta_1+\theta_2) + m_2 l_2^2 && \text{(B.4)}\\
&\quad +m_2 l_2 l_1 \cos(\theta_1)\cos(\theta_1+\theta_2) && \text{(B.5)}\\
H(2,1) &= H(1,2) && \text{(B.6)}\\
H(2,2) &= m_2 l_2^2 && \text{(B.7)}\\
C(1,1) &= 2 m_2 l_2 l_1 \dot{\theta}_2 \sin(\theta_1)\cos(\theta_1+\theta_2) && \text{(B.8)}\\
&\quad -2 m_2 l_2 l_1 \dot{\theta}_2 \cos(\theta_1)\sin(\theta_1+\theta_2) && \text{(B.9)}\\
C(1,2) &= m_2 l_2 l_1 \dot{\theta}_2 \sin(\theta_1)\cos(\theta_1+\theta_2) && \text{(B.10)}
\end{aligned}$$

$$-m_2\, l_2\, l_1\, \dot{\theta}_2\, \cos(\theta_1) \sin(\theta_1 + \theta_2) \tag{B.11}$$

$$C(2,1) = -2\,\dot{\theta}_1\, m_2\, l_2\, l_1\, \sin(\theta_1) \cos(\theta_1 + \theta_2) \tag{B.12}$$

$$+2\,\dot{\theta}_1\, m_2\, l_2\, l_1\, \cos(\theta_1) \sin(\theta_1 + \theta_2) \tag{B.13}$$

$$C(2,2) = -\dot{\theta}_1\, m_2\, l_2\, l_1\, \sin(\theta_1) \cos(\theta_1 + \theta_2) + \tag{B.14}$$

$$\dot{\theta}_1\, m_2\, l_2\, l_1\, \cos(\theta_1) \sin(\theta_1 + \theta_2) \tag{B.15}$$

$$K = \begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix} \tag{B.16}$$

$$G_{grav} = \begin{bmatrix} \frac{(m_1 l_1 + m_2 l_1) g \sin(\theta_1)}{\theta_1} + \frac{m_2 g l_2 \sin(\theta_1+\theta_2)}{(\theta_1+\theta_2)} & \frac{m_2 g l_2 \sin(\theta_1+\theta_2)}{(\theta_1+\theta_2)} \\ \frac{m_2 g l_2 \sin(\theta_1+\theta_2)}{(\theta_1+\theta_2)} & \frac{m_2 g l_2 \sin(\theta_1+\theta_2)}{(\theta_1+\theta_2)} \end{bmatrix} \tag{B.17}$$

Bibliography

[1] W.F. Ames. *Numerical Methods for Partial Differential Equations.* Academic Press, 1992.

[2] B.D.O. Anderson. Stability results for optimal systems. *Electron. Lett.*, 5:545, Oct 1969.

[3] H. Asada and J. Slotine. *Robot Analysis and Control.* Wiley, 1986.

[4] M. Athans. Class notes for multivariable control systems.

[5] R.H. Bartels and G.W. Stewart. Algorithm 432: Solution of the matrix equation $AX + XB = C$. *Commun. ACM*, 15:820–826, 1972.

[6] E. Bayo, P. Papadopoulos, J. Stubbe, and M.A. Serna. Inverse dynamics and kinematics of multi-link elastic robots: An iterative frequency domain approach. *Int. J. of Robotics Research*, 8(6):49–62, Dec 1989.

[7] R.W. Brockett. Feedback invariants for nonlinear systems. *IFAC Congress*, (6):1115–1120, 1978.

[8] A.E. Bryson and Y.C. Ho. *Applied Optimal Control.* Halstead Press, 1975.

[9] J.R. Bunch and D.J. Rose, (eds.). *Sparse Matrix Computations.* Academic Press, 1976.

[10] R.H. Cannon, Jr. and Schmitz E. Initial experiments on the end-point control of a flexible one-link robot. *Int. J.of Robotics Research*, 3(3):62–75, 1984.

[11] A.R. Collar. Some notes on Jahn's method for the improvement of approximate latent roots and vectors of a square matrix. *Quart. J. Mech. Appl. Math*, 1:145–148, 1948.

[12] W.A. Coppel. *Stability and Asymptotic Behavior of Differential Equations.* Heath, 1965.

[13] G. Dahlquist. Stability and error bounds in the numerical integration of ordinary differential equations. *Trans. Roy. Inst. Tech. (Sweden)*, 130, 1959.

[14] C.A. Desoer and H. Haneda. The measure of a matrix as a tool to analyze computer algorithms for circuit analysis. *I.E.E.E. Transactions on Circuit Theory*, 19(5):480–486, 1972.

[15] P. Dyer and S.R. McReynolds. *The Computation and Theory of Optimal Control.* Academic Press, 1970.

[16] B.A. Frances and G. Zames. On H_∞ optimal sensitivity theory for SISO feedback systems. *I.E.E.E. Transactions on Automatic Control*, 29:9–16, 1984.

[17] J.H. Ginsberg. *Advanced Engineering Dynamics.* Harper and Row, 1988.

[18] G.H. Golub and C.F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, 1983.

[19] G.C. Goodwin and K.S. Sin. *Adaptive Filtering Prediction and Control.* Prentice-Hall, 1984.

[20] J.D. Graham, R. Ravindran, and K. Knapp. Space manipulators – present capability and future potential. In *AIAA/NASA Conference on Advanced Technology for Future Space Systems*, pages 243–253, 1979.

[21] W. Hahn. *Stability of Motion.* Springer-Verlag, 1963.

[22] R.M. Hirschorn. Invertibility of multivariable nonlinear control systems. *IEEE Trans. on Aut. Control*, AC-24:855–865, 1979.

[23] L.R. Hunt, R. Su, and G. Meyer. Design for multi-input nonlinear systems. In *Differential Geometric Control Theory.* Birkhauser, 1983.

[24] A. Isidori. *Nonlinear Control Systems.* Springer-Verlag, 1989.

[25] B. Jakubczyk and W. Respondek. On linearization of control systems. *Bulletin de L'Academie Polonaise des Sciences, Serie des sciences mathematiques,* XXVIII:517–522, 1980.

[26] T. Kailath. *Linear Systems.* Prentice-Hall, 1980.

[27] S. Kenny. Private communications.

[28] D.E. Kirk. *Optimal Control Theory — An Introduction.* Prentice-Hall, 1970.

[29] D.L. Kleinman. On an iterative technique for Ricatti equation computations. *IEEE Trans. on Automatic Control,* AC-13:114–115, February 1968.

[30] P.V. Kokotovic, H.K. Khalil, and J. O'Reilly. *Singular Perturbation Methods in Control: Analysis and Design.* Academic Press, 1986.

[31] A.J. Krener. On the equivalence of control systems and the linearization of nonlinear systems. *SIAM J. Control,* 11(4):670–676, 1973.

[32] H. Kwakernaak and R. Sivan. *Linear Optimal Control Systems.* Wiley-Interscience, 1972.

[33] H. Kwakernaak and R. Sivan. The maximally achievable accuracy of linear optimal regulators and linear optimal filters. *IEEE Transactions on Automatic Control,* AC-17:79–86, 1972.

[34] W.H. Kwon, A.M. Bruckstein, and D.G. Byun. Receding horizon tracking control as a predictive control and its stability properties. *Int. J. Control,* 50:1807–1824, 1989.

[35] W.H. Kwon, A.M. Bruckstein, and T. Kailath. Stabilizing state-feedback design via the moving horizon method. *Int. J. Control,* 37:631–643, 1983.

[36] W.H. Kwon and A.E. Pearson. A modified quadratic cost problem and feedback stabilization of a linear system. *IEEE Transactions on Automatic Control*, AC-22:838–842, 1977.

[37] J. La Salle and S. Lefschetz. *Stability by Liapunov's Direct Method.* Academic Press, 1961.

[38] A.J. Laub. A Schur method for solving algebraic Ricatti equations. *IEEE Trans. on Automatic Control*, AC-24(6):913–921, December 1979.

[39] A.G.J. MacFarlane. An eigenvector solution of the optimal linear regulator problem. *J. Electron. Contr.*, 14:643–654, 1963.

[40] K. Martensson. On the matrix Ricatti equation. *Information Sciences*, 3:17–49, 1971.

[41] D.Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Trans. on Automatic Control*, AC-35(7):814–824, July 1990.

[42] O. Mayr. *The Origins of Feedback Control.* M.I.T. Press, 1970.

[43] R.H. Middleton and G.C. Goodwin. Adaptive control of time-varying linear systems. *I.E.E.E. Transactions on Automatic Control*, 33(2):150–155, 1988.

[44] D. Mitra and C. So Hing. Linear inequalities and p matrices with applications to stability of nonlinear systems. In *Proc. of the 5th Asilomar Conf. Circuits and Systems*, pages 179–188, 1971.

[45] P.J. Moylan and B.D.O. Anderson. Nonlinear regulator theory and an inverse optimal control problem. *IEEE Trans. on Automatic Control*, AC-18(5):460–465, October 1973.

[46] J.R. Munkres. *Analysis on Manifolds.* Addison-Wesley, 1991.

[47] Carlos E. Padilla. Nonlinear strain-displacement relations in the dynamics of a two-link flexible manipulator. Master's thesis, Department of Aeronautics and Astronautics, M.I.T., 1989.

[48] Carlos E. Padilla. *Performance Limits and Robustness Issues in the Control of Flexible Link Manipulators.* PhD thesis, Department of Aeronautics and Astronautics, M.I.T., 1992.

[49] J.-H. Park and H. Asada. Design and analysis of flexible arms for minimum-phase endpoint control. *American Control Conference*, 1990.

[50] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, 1988.

[51] H.H. Rosenbrock. A Lyapunov function with applications to some nonlinear physical systems. *Automatica*, 1:31–53, 1962.

[52] H.H. Rosenbrock. A Lyapunov function for some naturally ocurring linear homogeneous time-dependent equations. *Automatica*, 1:97–109, 1963.

[53] J.S. Shamma. *Analysis and Design of Gain Scheduled Systems.* PhD thesis, Massachusetts Institute of Technology, 1988.

[54] J.C. Simo and L. Vu-Quoq. The role of non-linear theories in transient dynamic analysis of flexible structures. *Journal of Sound and Vibration*, 119(3):487–508, 1987.

[55] J-J.E. Slotine. Sliding controller design for nonlinear systems. *Int. J. Control*, 40(2), 1984.

[56] J-J.E. Slotine. Putting the physics into control. *IEEE Control Systems Magazine*, 8(6), 1988.

[57] J-J.E. Slotine and J.A. Coetsee. Adaptive sliding controller synthesis for nonlinear systems. *International Journal of Control*, 43(6):1631–1651, 1986.

[58] J-J.E. Slotine and W. Li. On the adaptive control of robot manipulators. *Int. J. Robotics Research*, 6(3), 1987.

[59] J.J. Slotine and W. Li. *Applied Nonlinear Control.* Prentice-Hall, 1991.

[60] G. Stein and M. Athans. The LQG/LTR procedure for multivariable feedback control design. *IEEE Trans. on Automatic Control*, AC-32(2):105–114, 1987.

[61] R.F. Stengel. *Stochastic Optimal Control.* Wiley Interscience, 1986.

[62] P. Van Deventer. Pseudo-linear control methodology for nonlinear systems. Technical report, Massachusetts Institute of Technology, 1993.

[63] M. Vidyasagar. *Nonlinear Systems Analysis 2nd ed.* Prentice-Hall, 1993.

[64] V.I. Zubov: *Methods of A.M. Lyapunov and Their Application.* Noordhoff, 1964.